REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
		FINAL REPORT 01 Jun 93 - 30 Jun 96

4. TITLE AND SUBTITLE
(AASERT-92) Architectural Studies on Interfacing Parallel Optical Storage Systems with Processors

5. FUNDING NUMBERS
61103/D
3484/TS

6. AUTHOR(S)
Professor Sadik C. Esener

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Department of Electrical and Computer Engineering
University of California, San Diego
9500 Ollman Drive
La Jolla, CA 92093-0407

AFOSR-TR-96
0479

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFOSR/NE
110 Duncan Avenue Suite B115
Bolling AFB DC 20332-0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
F49620-93-1-0371

11. SUPPLEMENTARY NOTES

19961015 006

12a. DISTRIBUTION/AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE: DISTRIBUTION UNLIMITED

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Very large relational database machines require increased memory bandwidth and parallel processing to complete queries in a reasonable amount of time. Currently these machines achieve their capacity and throughput by employing an array of serial access disks and processors [1,2]. As these machines grow, however, so will their need for high capacity/bandwidth storage systems. 3D parallel access storage devices have the potential of being able to achieve enormous capacity (1 Tbit/cm^3)[3] as well as throughput (1 Tbit/sec) and seem well suited for this application. The bit-oriented 3D two-photon memory has a further feature of being able to be accessed in a number of directions. This report examines the potential performance of a bi-orthogonally accessed 3D two-photon memory for relational database operations with a data organization scheme particular to this approach. The favorable results of this study lead to another study in which the feasibility of building an optoelectronic database data filter to interface with the memory was considered.

14. SUBJECT TERMS

DTIC QUALITY INSPECTED 2

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	

Final Technical Progress Report

for

Architecture Studies on interfacing Parallel Optical Storage Systems with Processors

Sponsored by

Air Force Office of Scientific Research

Under Grant F-49620-93-I-0371

for Period 06/01/93 through 06/30/96

Grantee

The Regents Of the University of California, San Diego

University of California , San Diego

La Jolla CA 92093

Principal Investigators: Sadik C. Esener

(619) 534-2732

Program Manager: Dr. Alan Craig

(202) 767-4931

AASERT Evaluation Report

F49620-93-I-0371

Architecture Studies on interfacing Parallel Optical Storage Systems with Processors

Principal Investigator: Sadik C. Esener
University of California San Diego

a) Parent Award: F49620-93-I-0047
Architecture Studies in Parallel Optoelectronic Computing

b) Total amount of funding on parent award during the 12 months prior to 6/1/95: $114,284
0.5 full time equivalent graduate student

c) Total amount of funding on parent award during the 12 months after 6/1/96: $158,121
2 full time equivalent graduate student

d) Total amount of funding on AASERT for the 12 months after 6/1/94: $85,552 (Note: this is the total funding for the 3-year AASERT that was given on 6/1/93)

e) Brita Olson (124-64-6119) is a US citizen

Bi-orthogonally Accessed 3-D Two-photon Memory for Relational Database Operations

Brita Olson and Philippe J. Marchand

Very large relational database machines require increased memory bandwidth and parallel processing to complete queries in a reasonable amount of time. Currently these machines achieve their capacity and throughput by employing an array of serial access disks and processors [1, 2]. As these machines grow, however, so will their need for high capacity/bandwidth storage systems. 3D parallel access storage devices have the potential of being able to achieve enormous capacity (1 Tbit/cm^3) [3] as well as throughput (1 Tbit/sec) and seem well suited for this application. The bit-oriented 3D two-photon memory has a further feature of being able to be accessed in a number of directions. This report examines the potential performance of a bi-orthogonally accessed 3D two-photon memory for relational database operations with a data organization scheme particular to this approach. The favorable results of this study lead to another study in which the feasibility of building an optoelectronic database data filter to interface with the memory was considered.

1. BACKGROUND

As databases grow in size, they require increased memory bandwidth and parallel processing to complete queries in a reasonable amount of time. Currently very large database machines achieve their capacity and throughput by employing an array of serial access disks and processors [1, 2]. The bandwidth of these storage systems can be expanded, if the memories in the array are accessed in parallel. Mitkas and Berra have concluded that a relational database machine utilizing a single parallel access optical disk and a parallel optoelectronic processor has the potential of outperforming a relational database system based on an array of 30 serial access disks each with its own processor [4]. However, parallel access optical disks will not achieve capacities much greater than those of serial access optical disks.

3D parallel access optical memories are capable of achieving tremendous capacity in addition to throughput by storing data throughout the volume of the memory device as opposed to on the surface. Recent advances in both volume holographic [5] and 3D two-photon [6] page access optical memories show great promise for these high capacity/bandwidth storage devices. Mitkas and Irakliotis [7] have suggested their use in relational database systems.

In relational database machines the performance of operations depends not only on the memory bandwidth, but also on how the data is positioned and consequently accessed from the memory. The optimal data organization/access strategy for one database operation is not always best for another. A further property of the 3D two-photon memory is that pages of data can be retrieved from it in a number of directions; potentially allowing for a system in which a user can use different memory access approaches for different operations. Recently it has been suggested

that accessing a 3D two-photon memory in two orthogonal directions might be beneficial for relational database operations [7]. In this report, we describe a particular 3D data organization strategy for a relational database system which takes advantage of this accessing approach and evaluate its performance as a function of system parameters using the Wisconsin benchmark [8].

Our analysis shows that 3D parallel access optical memories, in general, are well suited for large relational database systems with low write requirements because of their potential capacity and throughput. The bi-orthogonal accessing approach used in conjunction with the proposed data organization strategy is found to be additionally advantageous. The time required to retrieve data for a projection operation, for example, is close to the optimum value. This is likely not be the case for a system with traditional page access and employing a data organization strategy optimized for another operation.

We begin this report with an introduction of relational database operations and their memory accessing requirements. Following this we give a brief description of the subset of data filtering operations from the Wisconsin benchmark which we used to evaluate performance. In Section 2, we provide a description of the bi-orthogonal memory itself and the proposed 3D data organization strategy. We also identify problems such as memory fragmentation which can lead to degraded performance in real systems. In Section 3, we analyze the potential performance of a bi-orthogonally accessed 3D two-photon memory, first considering the performance in an ideal system, and then we introduce the effect of the problems described in Section 2. In Section 4, we compare the potential performance of a 3D two-photon memory to that obtained with other proposed or existing relational database systems.

The favorable results of this study lead to another study in which the feasibility of building an optoelectronic data filter to interface with this memory was investigated. In particular, power consumption, volume and packaging requirements were considered.

2. OPERATION AND SYSTEM DESCRIPTION

2.1 Relational database operations

In relational databases, the data *records*, also called *tuples*, are grouped into sets called *relations*. All records belonging to a relation must be unique. Figure 2-1 depicts a relation comprised of student records. The columns are termed *fields* or *attributes*, and the rows are the records. Relational database operations can be categorized into two groups: set operations and relation-oriented operations [9]. Some of these operations are performed on data from a single relation. Others combine data from several relations. The result of an operation is always a relation.

If the data in a memory supporting bi-orthogonal access is arranged in a manner similar to that shown in Figure 2-1, a field or set of fields can be retrieved in one accessing direction without retrieving other fields that are not of interest. In the other accessing direction, a record or set of records can be retrieved without retrieving other records that are not of interest. These two particular accessing features which we term *record parallel* and *field parallel* access are particularly beneficial for relational database operations as they allow a user to better isolate the data desired for a given operation. We shall explore this further as we give examples of some relational database operations.

Relation 1: Student Records

student ID	Name	GPA	Major	Address
07839120	B. Jones	4.0	economics	9100 Regents Rd, SD CA 92037
07839121	T. Slash	3.7	engineering	221 Donex Ave, SD CA 92117
07839122	R. Kelsey	2.6	philosophy	3007 Ivy St, SD CA 92410
07839123	K. Smith	2.8	economics	102 Hall St, SD CA 92109
07839124	S. Jensen	3.2	biology	1765 Filbert St, SD CA 92116

record

field

Figure 2-1: An example of a relation which consists of student records. The rows of the table are termed *records* or *tuples*. The columns are *fields* or *attributes*.

A relation can be thought of as a set, and its records as elements. The set operations are the traditional set operations: union, difference and Cartesian product. These are not described here. Descriptions can be found in a standard database text [9]. The basic relation-oriented operations are: selection, projection, and join.

The selection operation, shown in Figure 2-2, is performed on data from a single relation. It involves choosing records from a relation based on the value of a particular field or set of fields. $<, >, \geq, \leq, =$ and Boolean complement are all valid selection operations. To perform this operation, records satisfying the query must be identified and retrieved. Frequently this can involve retrieving and examining all records which can take an enormous amount of time in large databases. If only a few records satisfy the selection query, the amount of data that needs to be retrieved for this operation can be significantly reduced if the operand is retrieved using field parallel access, and the records determined to satisfy the selection query are retrieved using record parallel access. In some instances, the value of the selection operand may directly correspond to its memory address. In this case, record parallel access might be preferable.

Selection Example: Which students have a GPA of 3.0 or higher?

student ID	Name	GPA	Major	Address
07839120	B. Jones	4.0	economics	9100 Regents Rd, SD CA 92037
07839121	T. Slash	3.7	engineering	221 Donex Ave, SD CA 92117
07839124	S. Jensen	3.2	biology	1765 Filbert St, SD CA 92116

Figure 2-2: Relation resulting from selection query STUDENT RECORDS(GPA $\geq$ 3.0). $<, >, \geq, \leq, =$ and Boolean complement are all valid selection operations.

With a projection operation, all the data from a relation which belongs to a single field or set of fields is extracted to form a new relation with fewer fields. True projection also involves duplicate removal; otherwise, the result would not be a relation. Once again, if the relation is

large, the process of extracting the data for this operation can be quite time consuming. If record parallel access was used to perform this operation all the data in the relation would have to be retrieved. If field parallel access were used instead, the amount of data that needs to be retrieved can be significantly reduced since the desired field or set of fields can be obtained directly.

Projection Example: **What Majors do students have?**

economics		economics
engineering	**duplicate removal**	engineering
philosophy	⟶	philosophy
economics		biology
biology		

Figure 2-3: Relation resulting from the projection operation STUDENT RECORDS(Major). With projection, a single field or several fields are extracted from a relation. Duplicate removal is necessary to guarantee that records in the resulting relation are unique.

A join operation combines relations. To illustrate join a second relation, depicted in Figure 2-4, is introduced which contains records corresponding to companies that might hire students. The result of a join operation is a subset of the Cartesian product of the two relations. Only those elements in the Cartesian product which satisfy the join query are in the join result. As with selection $<$, $>$, $\geq$, $\leq$, $=$ and Boolean complement are all valid join operations.

Relation 2: Companies hiring students

Company	Number of positions open	Target Major
ABC investments	0	econonics
Advanced Technology	0	engineering
XYZ BioTech	0	biology
American Banking	0	enonomics
Burger King	1	philosophy

Figure 2-4: Relation containing companies potentially hiring students.

Given the I/O and computational demands of the above operations, there are several approaches used to improve performance. Operations executed by a relational database machine are frequently *complex*, that is, they involve several basic operations. Selecting student-company pairs from the previous join example where companies have positions open is an example of a complex operation. Significant performance improvement can be gained by reordering

operations in complex operations [10]. In the above example, performing the selection first would reduce the complexity of the join.

Join Example:

Which students should be paired with which companies based on their majors?

student ID	Name	GPA	Major	Number of positions	Company
07839120	B. Jones	4.0	economics	0	ABC investments
07839121	T. Slash	3.7	engineering	0	Advanced Technology
07839123	K. Smith	2.8	economics	0	ABC investments
07839124	S. Jensen	3.2	biology	0	XYZ BioTech
07839120	B. Jones	4.0	economics	0	American Banking
07839123	K. Smith	2.8	economics	0	American Banking
07839122	R. Kelsey	2.6	philiosophy	1	Burger King

Figure 2-5: Example of a join operation. Company records are appended to student records if the student's major equals the company's target major.

Another way to improve performance, is to execute an operation in parallel. In addition to requiring parallel hardware, this approach requires increased memory bandwidth. In current systems this bandwidth is achieved with an array of serial disks, but it can also be achieved with parallel access storage.

2.2 Database data filter

Performance can also be gained if one filters out data deemed irrelevant to a particular query immediately when it is retrieved from secondary storage. A device providing this functionality is referred to as a data filter. This study assumes that an optoelectronic data filter is used in conjunction with the 3D two-photon memory. A number of other optoelectronic database data filters have been proposed [4, 11, 12]. The data filter considered in this study is assumed to support operations that can easily be done on the fly [13]: selection, and the operations projection and selection-projection, both without duplicate removal. To further improve performance, some data filters transform or reorganize the retrieved data in addition to reducing it. The complexity of an equijoin operation (a join where the comparison operation checks for equality), for example, can be reduced if the two relations are hashed on the field over which, they are being joined [14]. Data transformation and reorganization were not considered in this study. Filters providing this functionality could, however, be added.

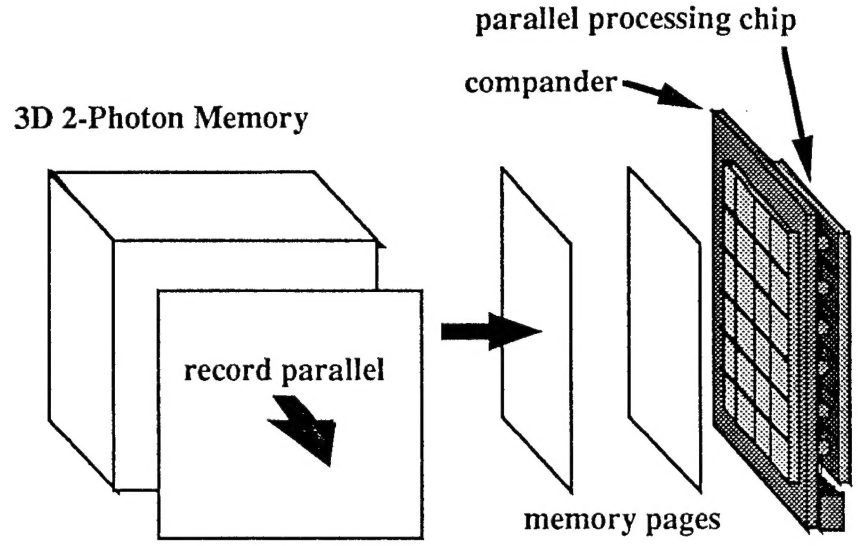

Figure 2-6: 3D optical memory with optoelectronic database data filter. The data filter consists of a compander which is flip-chip bonded onto an electronic processing chip. The compander is a modified CCD array which detects pages retrieved from the memory and interfaces this data with the electronic processing chip. The specifics of the compander and the electronic processing chip will be described in later sections.

The data filter that was assumed in this study is shown in Figure 2-6. It consists of a compander [15] which is flip-chip bonded onto an electronic chip capable of performing comparison and data masking operations in parallel. The compander is a modified CCD array which is used to detect the data from the memory and interface it with the electronic processing chip. It will be described in more detail in later sections.

2.3 Benchmark description

Numerous benchmarks have been proposed to evaluate database systems [16]. We rely on the Wisconsin benchmark, as it is commonly used to evaluate the performance of parallel relational database machines including optoelectronic ones. For clarity, portions of the benchmark that were used are briefly described. The benchmark allows databases of different sizes to be evaluated by scaling the size of relations. The operations considered in this study are performed on the same sized relation. However, other relations would exist in the memory at the same time.

The performance of the selection operation is examined for different selectivities. The *selectivity* of an operation refers to the number of records that satisfy a selection query. A *relative selectivity* of 10% means that 10% of the records in the relation satisfy the query. An *absolute selectivity* of 100 records means that 100 records satisfy the query regardless of the relation's size. The Wisconsin benchmark also requires that the performance of selection operations be measured using different types of *indexing* as well as *no indexing*. A relation with an *indexed* field is organized in some way (B-tree, hashing...etc.) according to the value of a particular field or set of fields; an index key is assigned to each record based on this value. With *clustered indexing*, the index key determines the physical location of the data. With *non-*

clustered indexing, the index key is used to determine the address of a record or set of records. Non-clustered indexing is not always preferable to no-indexing. Insertion of records has a much higher cost with indexing. Also, for selection operations with moderately high selectivity (1% - 10%), a crude scan of all the records (selection with no indexing) sometimes is comparable in performance or even preferable to non-clustered indexing [8]. The Wisconsin benchmark considers the performance of the different types of selection operations for a discrete set of selectivities: an absolute selectivity of one, and relative selectivities of 1% and 10%. It does not require selection with non-clustered indexing or no indexing for an absolute selectivity of one. It also does not require selection with non-clustered indexing for a selectivity of 10%.

In the sections that follow, the performance of various selection and projection operations is examined for a bi-orthogonally accessed 3D optical memory system. The specific operations considered in this study are the same as those in the Wisconsin benchmark except for the selectivity in selection operations, and the operand size in the projection operation, both of which were varied continuously instead of being assigned discrete values First, though, we briefly describe the memory and the proposed data organization scheme defining terms that shall be used for the duration of the report.

2.4 Bi-orthogonally accessed 3D two-photon memory

A parallel access 3D two-photon memory is a random access memory in which bits are suspended throughout the volume of a cube or rectangular solid, each occupying a unique physical location. Data are written via two photon excitation by intersecting two laser beams of different wavelength at any location within the memory. An entire page of bits can be written by intersecting a spatially modulated information beam with an addressing beam as shown in Report 4. A page written in this way can be retrieved by re-illuminating it with an identically positioned addressing beam, shown in Figure 2-7. The potential capacity of these memories is high (1 Tbit/cm^3) [3] because bits are stored in a volume as opposed to on a planer surface. Throughput is also high (1 Tbit/sec) because the pages themselves can contain 10^6 bits and be accessed in a relatively short amount of time. These particular memories can also be accessed in orthogonal directions, as shown in Figure 2-8. The ability to retrieve planes in two orthogonal directions is termed *bi-orthogonal* access.

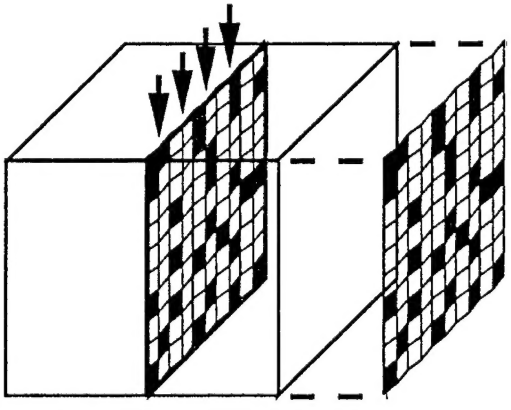

Figure 2-7: A page of bits can be retrieved from a 3D two-photon memory by illuminating it with a plane of light.

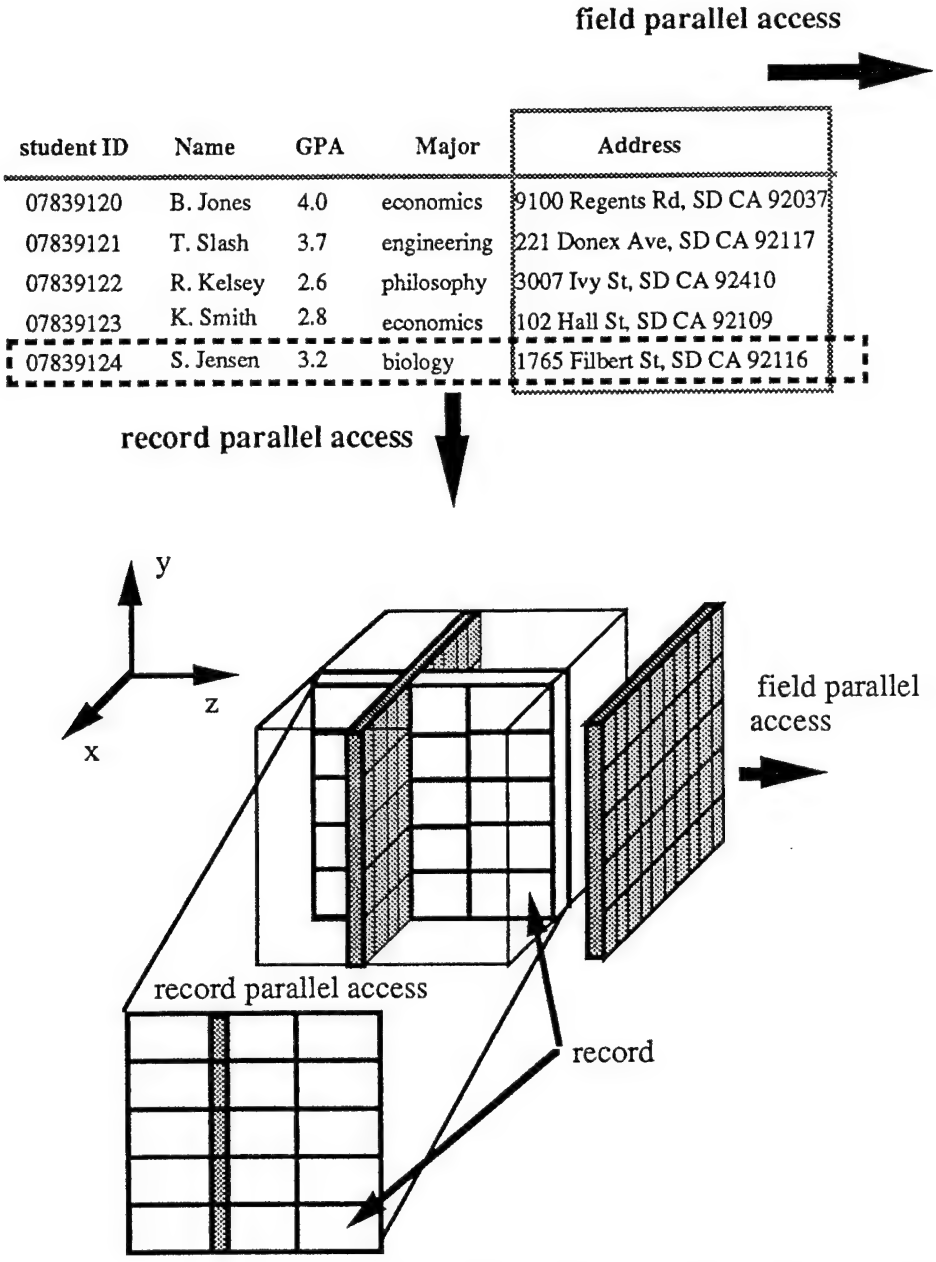

Figure 2-8: Data is organized in a bi-orthogonally accessed 3D two-photon memory such that complete records are mapped onto y-z planes. These records are arranged on these planes such that data belonging to the same field are on the same x-y plane. Accessing a y-z plane in parallel is referred to as record parallel access. Accessing an x-y plane is referred to as field parallel access.

2.5 Memory mapping

The data organization scheme that is proposed is explained with reference to Figure 2-8. Complete records are mapped onto the memory so that they are contained on a single y-z plane. These records are arranged on these planes such that data belonging to the same field or set of

fields are contained on the same x-y plane. Retrieving a y-z page containing complete records results in record parallel access. Retrieving an x-y page results in field parallel access. With record parallel access, a small set of records can be retrieved in parallel in one memory read retrieved without retrieving other records that are not of interest. With field parallel access, a particular field or set of fields can be retrieved without retrieving other fields that are not of interest.

Performance is improved by choosing the best accessing method for a given operation. Usually projection is best accomplished with field parallel access so that only the desired field or set of fields needs to be retrieved. This readout mode is also advantageous for certain selection operations as it provides an efficient means of scanning through a set of records to determine which records satisfy a particular selection query; only the operand on which the selection operation is based needs to be retrieved. If it is determined that a record satisfies the selection query, it can be retrieved in one page read using record parallel access. To read out a single record using field parallel access would require many more page reads. In the sections that follow, the effect that the two accessing approaches have on performance is examined.

Although physically the memory is roughly a cube, for reasons described in reference [17], the bits in the memory are not cubes; thus the memory has different linear capacities in different directions. In this report, however, the stored bits are represented as cubes and the memory as a rectangular solid with the long side corresponding to a direction with a higher linear data density. The memory is divided into bit cubes called *super-blocks*, shown in Figure 2-9. A super-block can be viewed as a sequence of pages that can be accessed randomly in either of two orthogonal directions. Each super-block contains M^3 bits and has M bits on a side; thus pages read from these super blocks contain M^2 bits. The time required to access a page is t.

The data organization scheme that is proposed is illustrated in Figure 2-10. Records are arranged on y-z planes, so that each x-y plane contains w bits of a record. It is assumed that w divides M. The parameter w affects both capacity and performance, and its effect will be examined throughout the paper. For example, it affects the time required to retrieve a record using field parallel access; approximately r/w page reads are required to retrieve a record of size r. We refer to the set of r/w x-y pages needed to contain a complete record as a *block*.

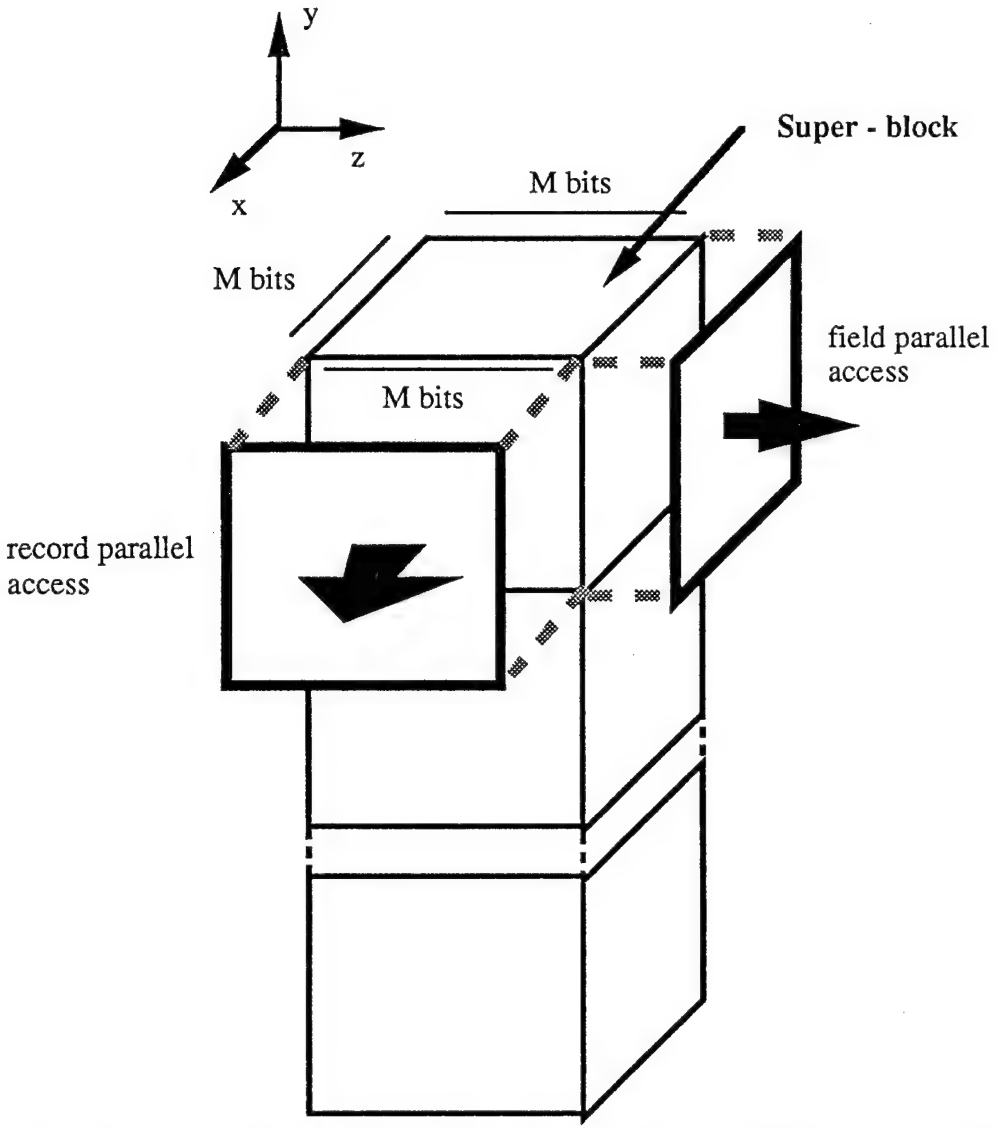

Figure 2-9: The memory is divided into bit cubes called *super-blocks*. A super-block can be viewed as a sequence of pages that can be accessed randomly in either of two orthogonal directions from any super-block in time t. Each super-block contains M^3 with M bits on a side; thus pages read from these super blocks contain M^2 bits.

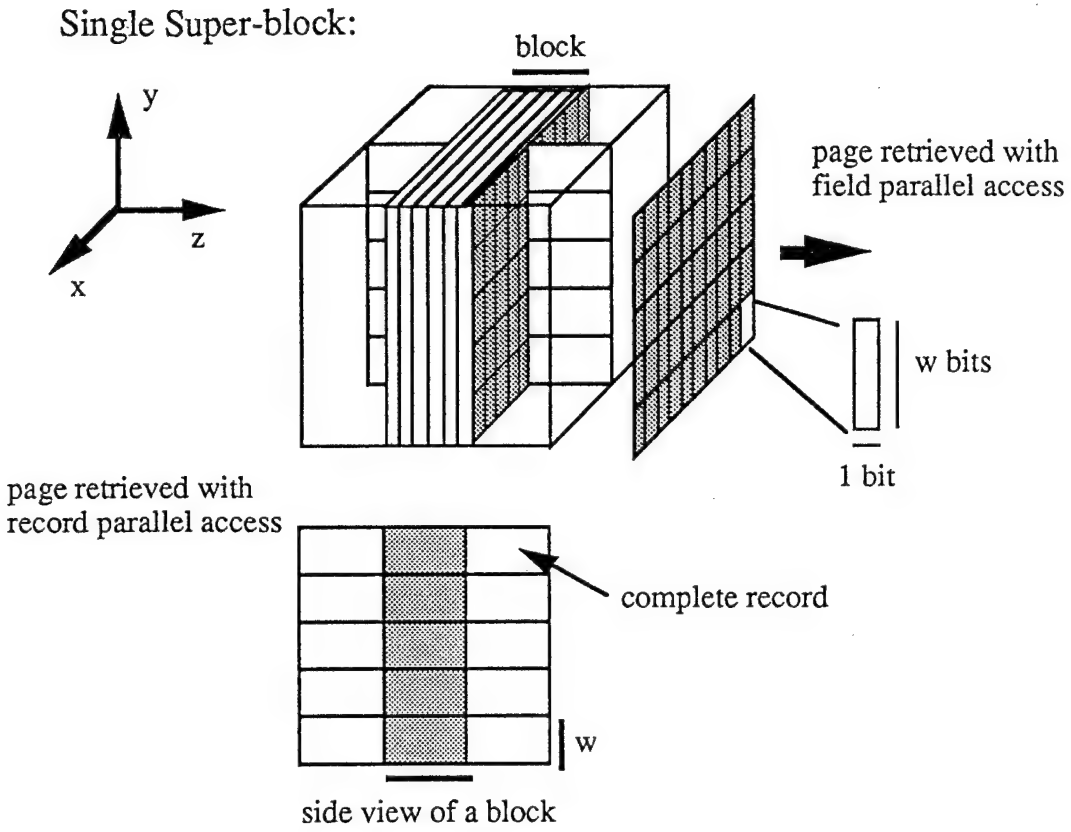

Figure 2-10: Records are mapped onto the memory such that they are contained on single y-z planes and the data on these planes are arranged so each page accessed in the field parallel direction contains w bits of a record. A complete record of size r can be accessed in one memory read utilizing record parallel access or in approximately r/w page reads using field parallel access. This set of x-y pages is referred to as a block.

With the proposed data organization scheme, there will sometimes be gaps in the memory which contain no data. This memory fragmentation problem will affect capacity and also performance, since pages retrieved from the memory will not always be full. Gaps can occur in two ways and are dependent on the value of w. It is assumed that data from multiple fields can be contained on the same field parallel page, and that the data in each group of w bits may belong to different records. The first type of gap is illustrated in Figure 2-11 and leads to memory fragmentation referred to as type I. This particular kind of memory fragmentation is likely to have little effect and is neglected in this study.

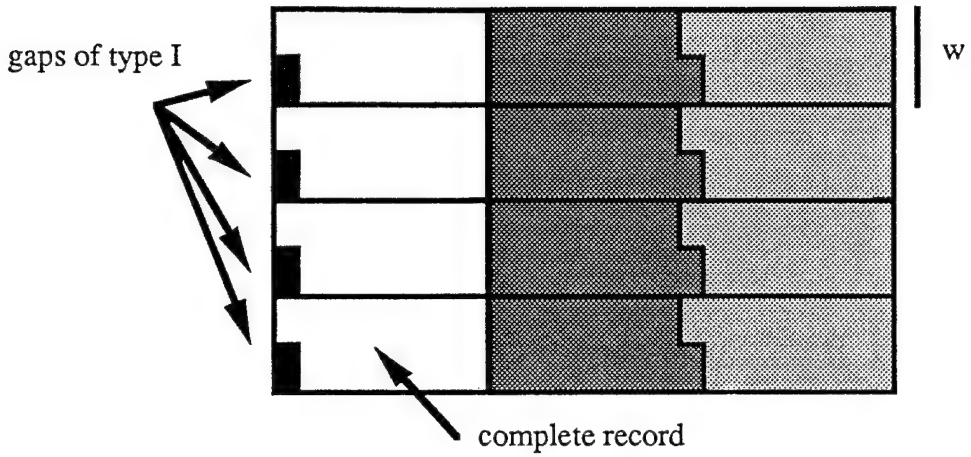

Figure 2-11: Page retrieved with record parallel access showing gaps leading to memory fragmentation of type I.

Another kind of memory fragmentation, termed type II, is more significant. It is further assumed that a record can always be retrieved with one memory read. The problem which can arise is explained with reference to Figure 2-12. This figure shows two pages retrieved with record parallel access, each employing a different record placement strategy, i.e., different values for w. In this figure w' and w" are chosen so that they divide M. However, P' and P" which are roughly r/w' and r/w", respectively, do not divide M. As a result, some memory on each record parallel page will not be used. Correspondingly, entire field parallel pages which are perpendicular to this will be empty as well. On average there will be P/2 such pages per super-block, where P ~= r/w is the number of pages in a block. The capacity penalty which results from this, on average increases when w is reduced. This fragmentation problem also affects data retrieval with record parallel access; with field parallel access, the empty pages could be avoided. We use parameter α to represent the data retrieval efficiency due to memory fragmentation when record parallel access is used. In Figure 2-12, it is the ratio of the area of the shaded region over the area of the entire page. Mathematically this can be expressed as:

$$\alpha = \left(\frac{r}{wM}\right)\left\lfloor\frac{Mw}{r}\right\rfloor \qquad (0.1)$$

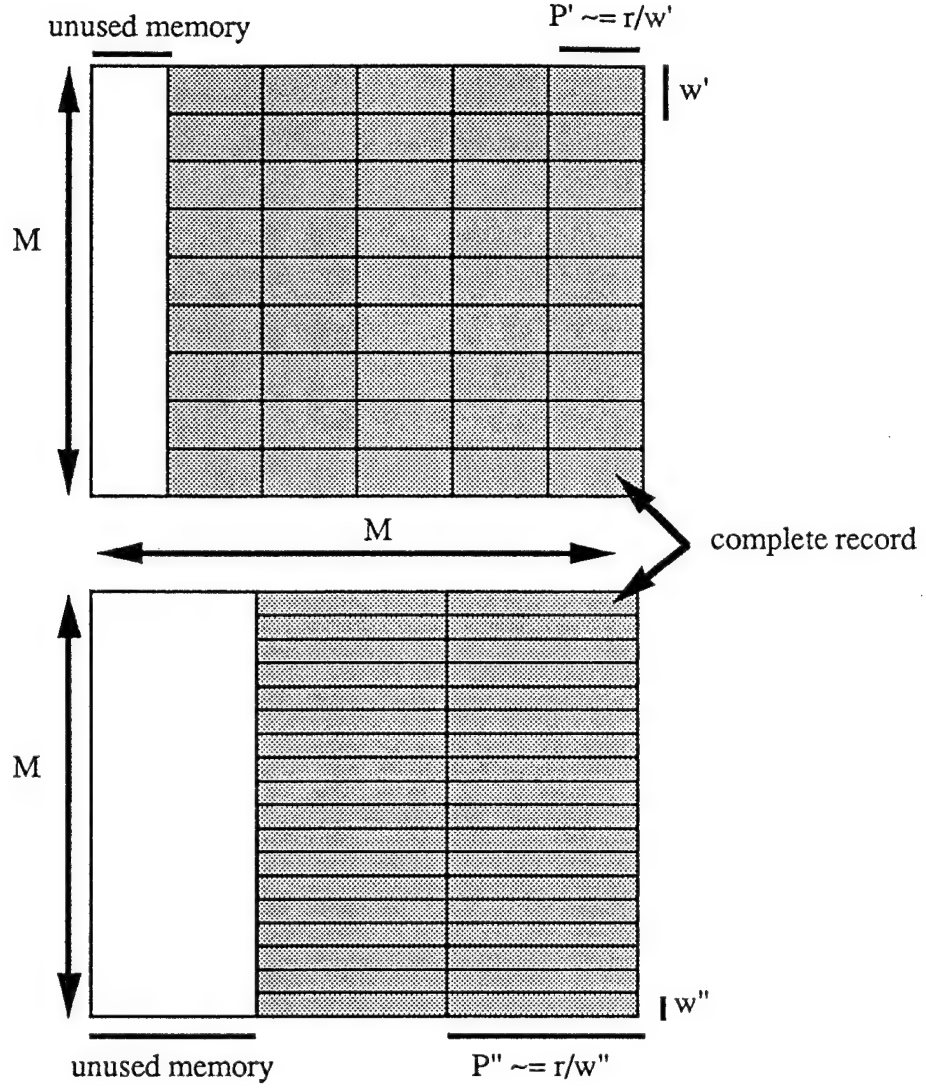

Figure 2-12: Memory fragmentation of type II can reduce capacity, and bit retrieval efficiency with record parallel access. Two pages retrieved with record parallel access are shown. w' and w" are chosen so that they divide M. It is likely that P' and P" won't divide M. As a result, some memory on each record parallel page will not be used. Correspondingly, entire field parallel pages which are perpendicular to this will be empty as well. The capacity penalty which results from this, on average increases when w is reduced.

Another factor that can have an even greater impact on performance is packing. A relation may not completely fill all the super-blocks in which it is contained; it is also likely to start and end in the middle of a super-block. As a result of this, the first and last super-blocks will usually contain other relations or be empty. Records residing in these partially filled super-blocks can be placed such that they fill field or record parallel pages first. This is shown in Figure 2-13. In general if field parallel pages are filled first, the time to perform operations using record parallel access will approximately increase by a factor of 1/B, where B is the minimum number of super-blocks required for the operation. This is because on average two additional

half super-blocks of data will have to be read that do not contain the relation of interest. The time required to perform operations using field parallel access is not affected. If, on the other hand, record parallel pages are filled first, the record parallel access time is not affected, but the field parallel access time is increased by a factor of 1/B. The effect of adverse packing decreases for larger relations, but can be significant for small relations when there are few super-blocks and B is small.

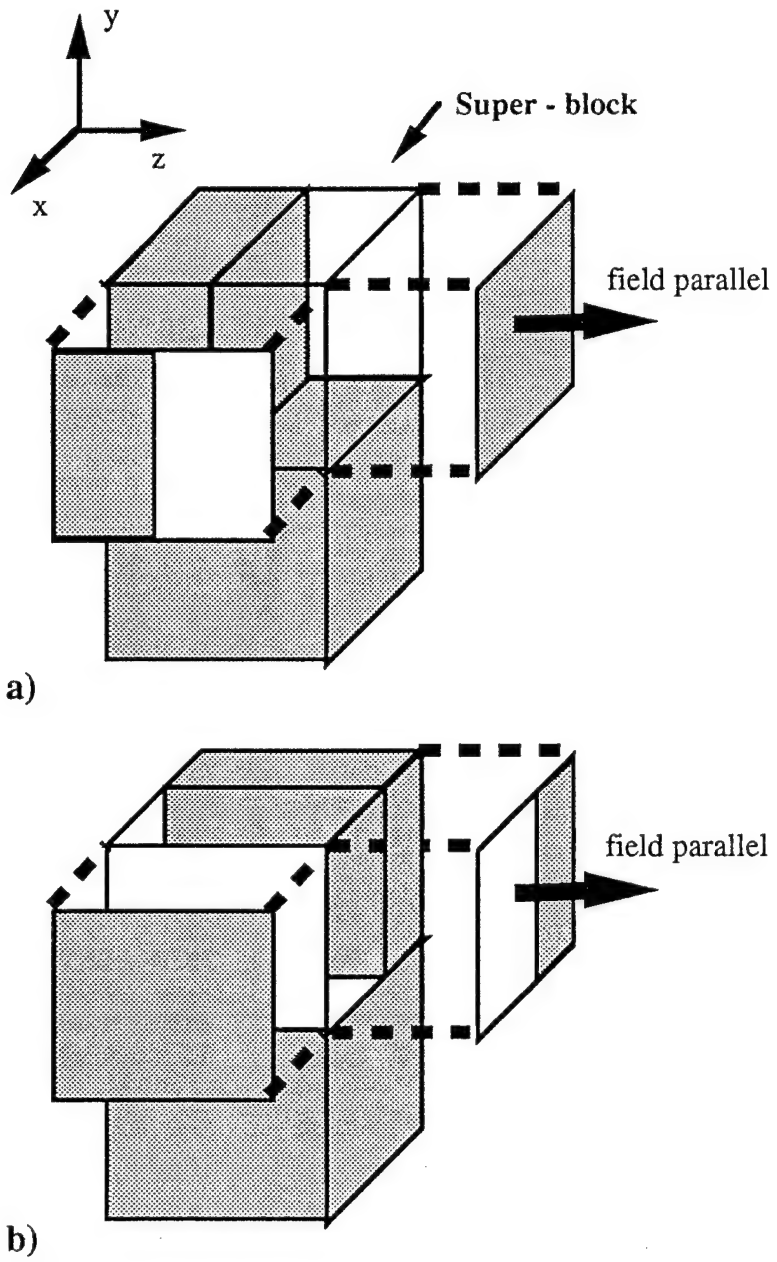

Figure 2-13: If field parallel pages are packed first, a), some pages retrieved with record parallel access will contain data from other relations. If record parallel pages are packed first, b), field parallel access can be adversely affected.

3. PERFORMANCE STUDY

3.1 Performance study parameters

In this section we analyze the potential performance of a bi-orthogonally accessed 3D optical memory for a subset of relational database operations commonly used in data filters. The particular operations that were considered were projection and selection with and without indexing. We begin by first looking at the effect that w and the accessing method have on performance in an ideal situation, neglecting the effect of memory fragmentation and packing. Later we consider the effect that these two factors can have.

For clarity the parameters and values that are used in the equations and graphs in this and the following section are summarized in Table 3-1. When numerical results are given it is assumed that the relation contains 10^6 records. This particular relation size was chosen to facilitate comparison with other existing or proposed relational database machines. A single two photon memory could hold a much larger sized relation.

3.2 Ideal projection

The projection operation requires that a field or set of fields be extracted from a relation to form a new relation. This operation also requires duplicate removal. In our analysis, we only considered the time that would be required to retrieve the data for the operation not the time for duplicate removal as it would have required the analysis of a complete database system. The times given for projection in this section, therefore, cannot be compared to times obtained for other systems using the Wisconsin benchmark, since the benchmark which requires duplicate removal.

The projection operation is almost always best performed with field parallel access since only the desired field or set of fields needs to be retrieved. With record parallel access, the entire relation always needs to be retrieved. The time required to retrieve the data necessary for a projection operation with record parallel access is given below. It is simply the time required to retrieve a single page, t, multiplied by the number of record parallel pages needed to store the relation. In this equation and equations that follow, "$\lfloor\ \rfloor$"denotes the integer less than or equal to the operand in brackets and "$\lceil\ \rceil$" denotes the integer greater than or equal to the operand in brackets.

$$t\left\lceil \frac{R\,r}{M^2} \right\rceil \qquad (0.2)$$

The time required to retrieve the data using field parallel access is given in equation 0.3, assuming the desired field or set of fields is p bits. The first two terms represent the time required to retrieve the p bits from a single block in the memory. The last term is the number of blocks needed for the relation.

$$t\left\lceil \frac{p}{w} \right\rceil \left\lceil \frac{R\,w}{M^2} \right\rceil \qquad (0.3)$$

parameter	description	value or expression
R	number of records in relation	10^6
r	record size	(208×8) bits
M^2	memory page size	$1024^2 \cong 10^6$ bits
M	number of memory pages in a super-block	1024
t	time to access a memory page	10 usec
w	field parallel word size	variable
α	page retrieval efficiency due to fragmentation of type II	$\left(\frac{r}{Mw}\right)\left\lfloor\frac{Mw}{r}\right\rfloor$
B	minimum number of super-blocks required to store the relation	$\left\lceil\frac{rR}{\alpha M^3}\right\rceil \cong 2$
N	number of records satisfying the selection query	variable
f	operand size for selection operation	32 bits
p	operand size for projection operation	variable

Table 3-1: Variables used for performance calculations

The time required to retrieve the data for a projection operation with the two accessing methods is plotted in Figure 3-14 as a function of the operand size. The operand size is augmented in byte increments even though this could physically correspond to reading out fractions of fields. The superior performance of field parallel access can clearly be seen. For small operand sizes and small values of w, the time required to retrieve the data with field parallel access can require a factor of 50 less time than with record parallel access. The jagged appearance of the field parallel trace results from the fact that the operand size p is not always a multiple of w bits. The time required to perform this operation with field parallel access is minimized when p is a multiple of w which is more likely to occur when w is small.

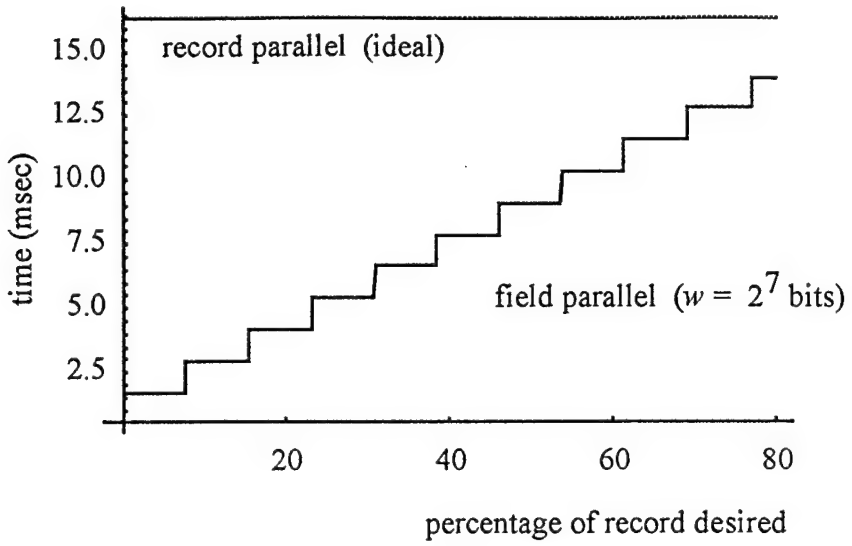

Figure 3-14: Time to retrieve the data required for a projection operation. With field parallel access only the desired field or set of fields needs to be retrieved. With record parallel access, the entire relation needs to be retrieved. This graph neglects seek time and the effects of packing and memory fragmentation.

3.3 Ideal selection with clustered indexing

The performance of selection with clustered indexing was also considered. With this operation, the location of the records is known a priori, and the records satisfying the selection criterion are assumed to be adjacent. Two different data ordering schemes are assumed. With record parallel access, consecutive records are on the same record parallel page, while with field parallel access, consecutive records are on the same field parallel page. The Wisconsin benchmark requires that the time required to return and format the result to the user be included for this particular selection operation. This overhead is not included in this analysis -- only the time required to retrieve the desired records.

In principal it is best to perform this operation with record parallel access. In practice, when one includes the effect of memory fragmentation and packing, this may not be the case. The time to perform this operation with record parallel access is given below, assuming that N is the number of records which satisfy the selection query. The time is roughly equal to the page access time, t, multiplied by N divided by the number of records on a page ($\sim M^2/r$). There is a good chance that the set of consecutive selected records will start and end in the middle of a record parallel page, and that an additional page read will be required. This is taken into consideration in equation 0.4.

$$t\left((N-1)\left\lfloor\frac{M^2}{r}\right\rfloor^{-1}+1\right) \qquad (0.4)$$

The expression for the time required to perform this operation with field parallel access is similar and is given below. With field parallel access, r/w page reads are required to retrieve a

complete record. This set of pages as mentioned earlier is referred to as a block. The time to perform this operation with this form of access is roughly equal to the amount of time required to readout a block, t r/w, multiplied by the number of blocks which would be needed to contain the N selected records: $N\,w/M^2$. Equation 0.5 also includes the probability that an additional block may need to be retrieved.

$$t\left(\frac{r}{w}\right)\left(\frac{(N-1)\,w}{M^2}+1\right) \qquad (0.5)$$

The time required to perform selection with no indexing is plotted in Figure 3-15 as a function of relative selectivity for the two different accessing modes. As mentioned earlier, the set of contiguous selected records will occupy a certain minimum number of pages in the case of record parallel access or blocks in the case of field parallel access and be likely to start and end in the middle of a page/block, thereby requiring an additional page/block read. The overhead of having to read an additional block with field parallel access is greater than the overhead of retrieving an additional page with record parallel access. Thus, under ideal conditions, this operation is best performed with record parallel access. With field parallel access, the overhead is larger when there are more pages in a block (when w is small). Performing very low selectivity operations, such as retrieving a single record, is also best performed with record parallel access. With this strategy only one page read is necessary, while with field parallel access an entire block would have to be retrieved.

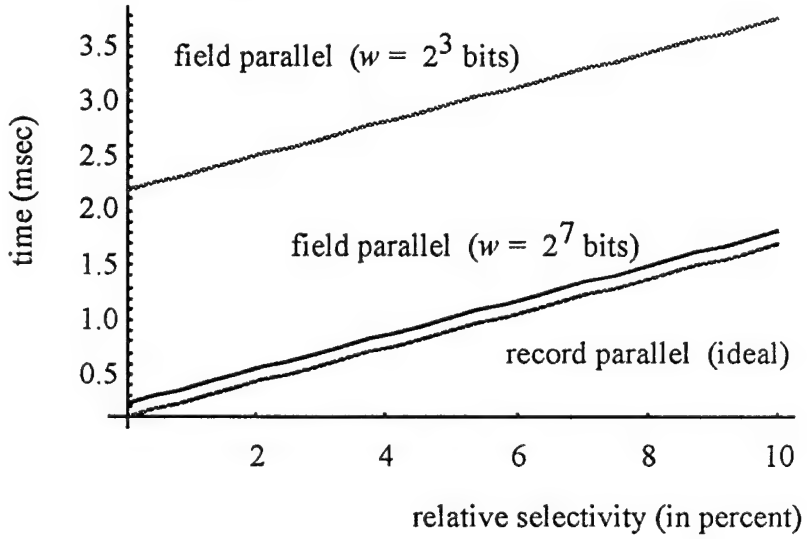

Figure 3-15: Time to perform selection with clustered indexing neglecting the effect of seek time, packing and memory fragmentation.

3.4 Ideal selection with no indexing

The performance of selection with no indexing was also considered. With this operation, the locations of the records satisfying the selection query are not known a priori, so all the records have to be searched to determine which ones satisfy the selection query. Records which

satisfy the query would then be retrieved. It is assumed that selected records are uniformly distributed throughout the memory.

Ideally the accessing method that should be used for this operation depends on the selectivity of the operation. Very low selectivity operations should be performed by searching using field parallel access, and retrieving selected records with record parallel access. With moderately high selectivity operations, all pages will have to be retrieved so either field or record parallel access could be used.

In general, if this operation is performed using record parallel access, all record parallel pages containing the relation will have to be retrieved so that each record can be examined to determine which ones satisfy the selection query. Thus, the time required to perform this operation with record parallel access is the same for all selectivities, and is the same as the time to retrieve records for a projection operation with record parallel access.

w	T_{search} (in msec)
2^3 bits	.32
2^5 bits	.31
2^6 bits	.62
2^7 bits	1.23

Table 3-2: Effect of w on T_{search}. T_{search} increases when w is larger than f. In this example f is 32 bits.

This operation can also be performed with field parallel access or by combining field and record parallel access. The first accessing approach is termed *field-field* and the later *field-record*. With the two approaches, the search part of the operation is performed using field parallel access; only the operand containing field(s) is/are retrieved. Records satisfying the selection query are read out using record parallel access (for the field-record approach) or using field parallel access (for field-field). The search part of the operation requires time T_{search} ; an expression is given in equation 0.8 and tabulated for different values of w in Table 3-2. It is assumed that $\lceil f/w \rceil$ field parallel page reads are needed for every block, where f is the size of the selection operand. (There is a probability that it could require one more page read than this.) The searching time is inefficient when w is a much larger than f. The time required to retrieve the selected records is referred to as the readout time. The time to read out records satisfying the selection query is denoted as T_{ror} if record parallel access is used and as T_{rof} if field parallel access is used. With this notation, the expression for the average time to perform ideal selection with no indexing and field-record access is given below assuming that N records satisfy the selection query.

$$T_{search} + T_{ror} \qquad (0.6)$$

where,

$$T_{ror} = T + t\left\lceil \frac{R}{\left\lfloor M^2/r \right\rfloor} \right\rceil \left(1 - \left(1 - \left\lceil \frac{R}{\left\lfloor M^2/r \right\rfloor} \right\rceil^{-1} \right)^N \right), \qquad (0.7)$$

and

$$T_{search} = t\left\lceil \frac{f}{w} \right\rceil \left\lceil \frac{R\,w}{M^2} \right\rceil. \qquad (0.8)$$

In the equation for T_{ror}, the quantity raised to the N^{th} power is the probability that a page is empty given N records are selected. One minus this quantity raised to the N^{th} power is the probability that a page contains at least one selected record and needs to be read out. It should be noted that the selection operand is readout twice with this approach during both the searching and readout portions of the operation. The first two terms in the expression for T_{search} represent the time required to read out the operand from a single block. The second term is the number of blocks in the memory.

The average time to perform selection with no indexing using field-field access is given below. The expression for T_{rof} is very similar to the expression for T_{ror}. The quantity raised to the N^{th} power is the probability that a block is empty given that N records are selected. One minus this quantity raised to the N^{th} power is the probability that a block has a selected record on it and will need to be retrieved. It is assumed that the selection operand does not have to be read out twice during the search and readout portions of the operation. This is taken into consideration with the term multiplied by T_{rof} in equation 0.9.

$$T_{search} + \left(1 - \left(\frac{w}{r} \right) \left\lceil \frac{f}{w} \right\rceil \right) T_{rof} \qquad (0.9)$$

where,

$$T_{rof} = t\left(\frac{r}{w} \right) \left\lceil \frac{R\,w}{M^2} \right\rceil \left(1 - \left(1 - \left\lceil \frac{R\,w}{M^2} \right\rceil^{-1} \right)^N \right). \qquad (0.10)$$

The time to perform ideal selection with no indexing is plotted vs. selectivity in Figure 3-16 for the three access modes. The search time for the field-record plot is assumed to be .31 msec. The time to perform this operation saturates for the two field parallel based accessing approaches. With field-record access, this occurs when it is likely that every record parallel page will have a selected record on it and will need to be read out. At this point the fraction of records satisfying the selection criterion is roughly equal to one divided by the number of records on a page. For field-field access the saturation happens when it is probable that every block needs to be read out: when the fraction of records satisfying the selection criterion is approximately equal

to one divided by the number of records in a block; when w is smaller the saturation occurs sooner.

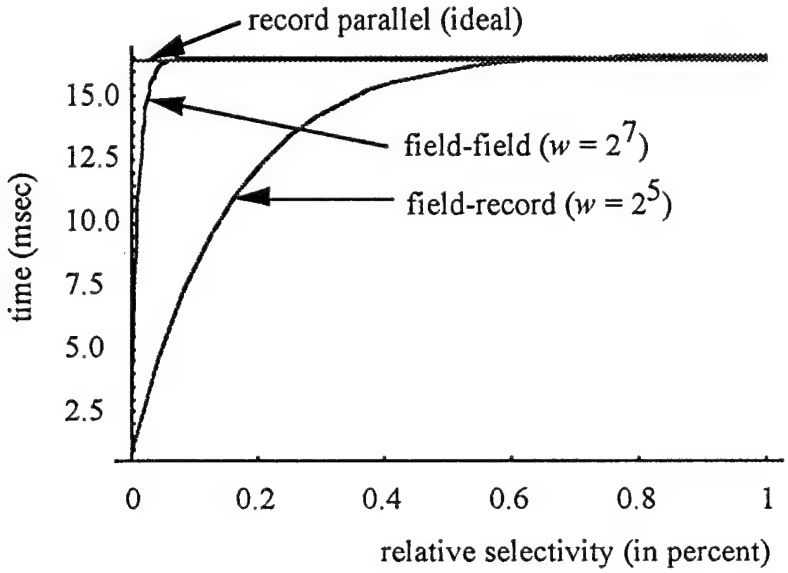

Figure 3-16: Time to perform selection with no indexing neglecting the effect of seek time and packing. For record parallel and field-record access the effect of memory fragmentation is not included. The search time for field-record access is assumed to be .31 msec.

The field-record parallel readout approach can be disastrous depending on how the search and readout parts of the operation are carried out with respect to each another. This problem is explained with reference to Figure 3-17 which shows a single super-block. On this diagram, the $\lceil f/w \rceil$ field parallel page reads required to scan through a single block, don't provide information about the other records on the record parallel page that is completely visible. $(b-1)\lceil f/w \rceil$ additional field parallel pages would need to be retrieved, to determine the total number of records satisfying the selection query on the visible page, where b is the number of blocks in a super-block. To improve the performance of this operation all blocks in a super-block should be searched first before attempting to read out selected records using record parallel access. If instead the records are retrieved after each block is searched, each record parallel page may have to be readout b times -- once for each block.

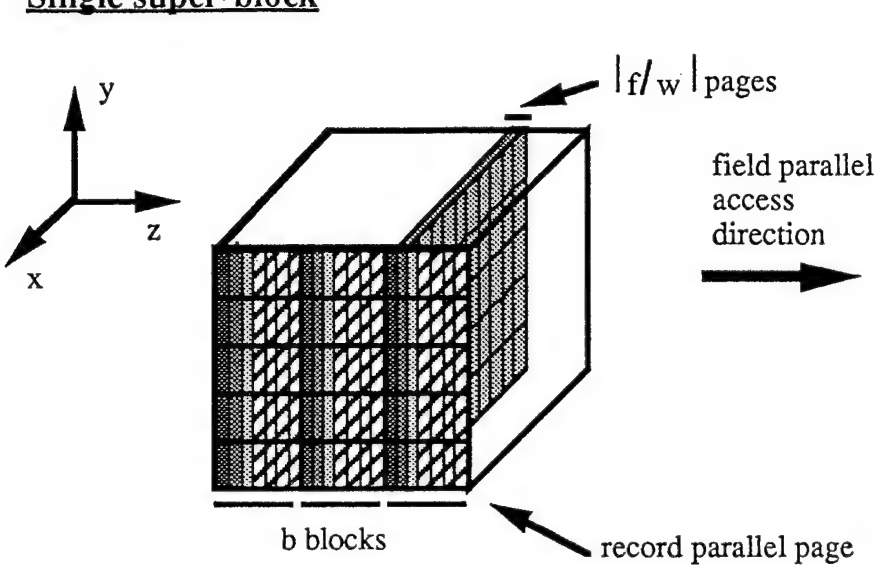

Figure 3-17: For selection with no indexing performed using field-record access, $\lceil f/w \rceil$ field parallel page reads are required per block in the search part of the operation. These pages reads yield little information about the record parallel page that is completely visible and those pages behind it. Additional field parallel pages would need to be retrieved, to determine the total number of records selected on the visible page, where b is the number of blocks in a super-block. With this in mind, if this operation is performed with field-record access, all blocks in a super-block should be searched first using field parallel access before attempting to read out the selected records using record parallel access.

3.5 Ideal selection with non-clustered indexing

With the bi-orthogonally accessed 3D two-photon memory, non-clustered indexing would only be preferable to selection with no indexing if it could eliminate more page reads. A lower bound on the time to perform selection with non-clustered indexing is the time that it takes to read out pages or blocks with selected records from the memory (assuming no page is read out twice). This bound neglects the time to determine where the records are. The difference between the two operations is the search time, T_{search}, which is generally quite small. Selection with non-clustered indexing would only be advantageous for extremely low selectivity operations, when the search time is comparable to the readout time or larger.

3.6 The effect of memory fragmentation

At this point we a have considered the performance of a bi-orthogonally accessed 3D memory under ideal conditions. In real systems there will be practical issues which can cause the performance to deviate from the ideal. On such problem is memory fragmentation, which occurs when records fit poorly onto record parallel pages. As described in the previous section, this problem reduces the usable memory capacity and the bit retrieval efficiency by a factor of α. Correspondingly the time to perform operations or portions of operations using record access is

increased by roughly a factor of $(1-1/\alpha)$. This factor will generally be greatest when w is small. Table 3-3 lists values obtained for α and $1/\alpha$ for various values of w. When w is eight bits the time required to perform this operation with record parallel access is increased by 23%, since $1/\alpha$ is equal to 1.23. However, when w is larger, there is very little inefficiency.

size of w	α	$1/\alpha$
2^3 bits	.81	1.23
2^5 bits	.96	1.04
2^6 bits	.99	1.01
2^7 bits	.99	1.01

Table 3-3: Effect of w on α. The overhead for storing a record increases when w is small due to memory fragmentation of type II.

This particular problem in some instances can make field parallel access preferable when under ideal conditions record parallel access would have been. For selection with clustered indexing, for example, field parallel access with $w = 2^7$ bits can be preferable to record parallel access if w is small and the selectivity is high. This is also the case for selection with no indexing for high selectivites.

3.7 The effect of the packing strategy

Packing can have an even greater impact on performance than memory fragmentation. As described in the previous section, packing on average extends the time required for an operation by 1/B, where B is the minimum number of super-blocks required to store the relation, depending of the accessing technique used. Thus if record parallel pages are packed first, operations or portions of operations performed with field parallel access will on average require 1/B more time to complete. However, the time to perform operations with record parallel access will not be affected. Likewise if record parallel pages are packed first the time for field parallel access will similarly increase, but the time to perform this operation with record parallel access will not be affected. In this study, B is equal to two. Thus the average time to perform operations can be increased by 50%, depending on the accessing technique used. While the effect of adverse packing decreases for larger relations, it increases for smaller relations.

3.8 Bit retrieval efficiency

A metric termed bit retrieval efficiency was devised to measure the optimality of operations performed with the bi-orthogonally accessed 3D optical memory. It is defined as the ratio of the minimum number of bits required for an operation to the average number of bits that are retrieved. It reflects the utilization of memory bandwidth and the random access capability of the memory device. The average number of bits that are retrieved for a given operation is determined by multiplying the average number of pages retrieved by the number of bits on a page. The average number of pages is computed by dividing the time to perform the operation by t, the page access time.

When projection is performed with field parallel access the bit retrieval efficiency is roughly one, the most optimal. It decreases when w is large, particularly if the desired field is

very small. It also decreases if record parallel pages are packed first, by roughly a factor of B/(B+1).

For selection with clustered indexing, the minimum number of bits that would have to be retrieved for the operation is: rN. The bit retrieval efficiency for selection with clustered indexing is plotted vs. selectivity for the different access modes in Figure 3-18 neglecting the effect of packing. For most selectivities, the bit retrieval efficiency is quite good and approaches one (record parallel first) with increasing selectivity. Only for very low selectivity operations is this operation inefficient, when, for example, an entire page or block has to be retrieved when only one record is desired.

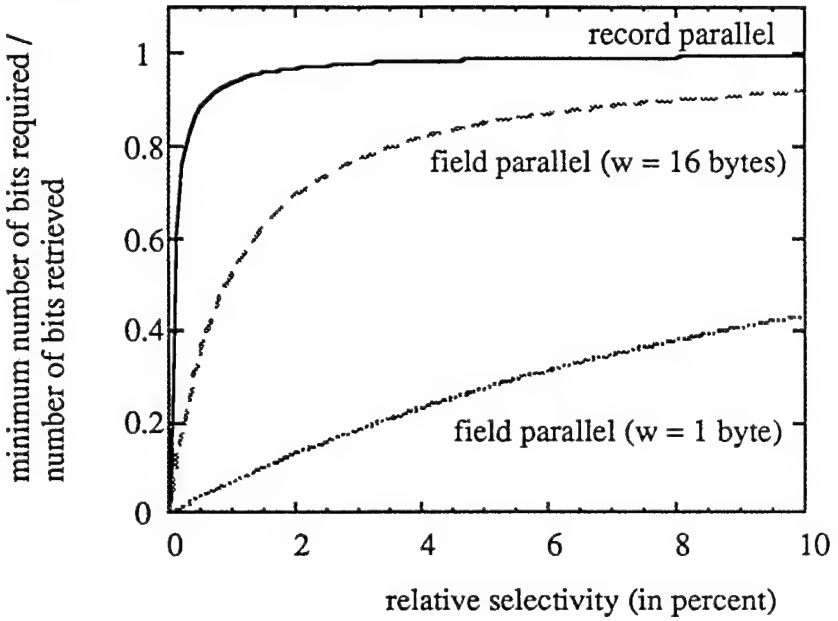

Figure 3-18: bit retrieval efficiency for selection with clustered indexing. This neglects the effect of memory fragmentation, seek time and packing.

The minimum number of bits required for selection with no indexing can be expressed as: $fR+(r-f)N$. The first term reflects the searching and the second the read out. It assumes the selection operand is not read out twice. The bit retrieval efficiency for this operation is plotted in Figure 3-18 neglecting the effect of packing, and assuming that selected records are distributed randomly throughout the memory. Performing this operation with record parallel access is inefficient in all but extremely high selectivity operations, because all record parallel pages have to be retrieved. For the two field parallel accessing based approaches (field-field and field-record), the searching part of this operation is performed efficiently. It is the readout part of the operation that is inefficient. If selected records are uniformly distributed throughout the memory, it is likely that pages and blocks will only have a few selected records on them, but will need to be readout. The efficiency improves as the number of selected records per page (or block). For the selectivity range shown in the graph, field-record access is most efficient for very low selectivity operations. In this selectivity range, the searching, which is quite efficient, dominates, and the readout inefficiency is not noticed.

The bit retrieval efficiency with a bi-orthogonally accessed 3D optical memory is in general quite good. This is because the accessing feature improves the random access capability of the memory. The graph of the bit retrieval efficiency, Figure 3-18, for selection with no indexing, provides insight into what future developments in memory technology can improve the performance of database systems. With this particular operation, it is likely that a page or block will have a small number of selected records on it and will need to be read out. A bi-orthogonally accessed 3D optical memory would outperform the one studied here if its page size were reduced and its bandwidth kept the same.

4. PERFORMANCE COMPARISON WITH OTHER SYSTEMS

In this section the performance of the 3D bi-orthogonally accessed memory is compared with the performance of the GAMMA [1] parallel database machine and the projected performance of PHOEBUS [4], an optoelectronic parallel database machine based on parallel readout optical disk technology. These two systems are described briefly below.

GAMMA is a parallel relational database machine that was built at the University of Wisconsin. It is based on a shared-nothing architecture and has 32 Intel 386 processors each with 8 Mbytes of memory and its own 330 Mbyte disk and controller. The processors are connected with an Intel iPSC/2 hypercube network. Data is distributed across the disks using *horizontal-partitioning*. With this data organization scheme, individual records are stored together in the same file, but records belonging to a relation are distributed across all the disks. This organization allows operations to easily be executed in parallel on all 32 processors. It should be noted that the benchmark was run on a system with 30 processors and disks.

PHOEBUS is a proposed optoelectronic database machine which utilizes parallel readout optical disks. In this system complete records are arranged on radial strips on the surface of a Magnetooptical optical disk and are read out in parallel one at a time. The polarization rotation of retrieved ONEs and ZEROs from the memory is $+\theta$ degrees and $-\theta$ degrees respectively. For selection with no indexing which checks for equality (Only this particular selection operation is described and supported), records are retrieved from the disk; fields other than the one involved in the selection operation are filtered out with a "mask". The desired field is then positioned on the operand holding SLM. A 1D polarization rotating SLM is loaded with the operand. A ZERO on the SLM will cause the polarization of a bit of light to be rotated by $+\theta$ degrees and a ONE by $-\theta$ degrees. If bits on the SLM agree with those retrieved from the memory, the polarization of the retrieved bit will be rotated such that it is 0 degrees. Bits not agreeing will be rotated to + or - 2θ degrees. An analyzer is placed after the SLM which functions to filter out bits with 0 degree polarization rotation. After the analyzer, bits that are ON signify bit differences. The light at the output of the analyzer is focused onto a single detector. If a value other than ZERO is detected, it signifies that the field from the retrieved record did not equal the operand.

Table 4-4 gives the projected performance of the 3D two-photon memory based data filter for selection with no indexing using the Wisconsin benchmark with w equal to 2^5 bits. The time to perform this operation with 1% and 10% selectivity is the same because all pages and

blocks are very likely to have at least one selected record on them. The selection times were calculated using the equations given earlier, and the values listed in Table 3-1.

Record parallel pages packed first:

Type of access	Time (in msec)
record parallel	16.45
field-field access	24.19
field-record access	16.92

Field parallel pages packed first:

Type of access	Time (in msec)
record parallel	24.67
field-field access	16.13
field-record access	16.75

Table 4-4: Time to perform selection with no indexing with a selectivity of 1% and 10% using the Wisconsin benchmark. w equals 2^5 bits.

Table 4-5 lists the times to perform selection with no indexing for GAMMA along with the projected times for PHOEBUS and the bi-orthogonally accessed two-photon memory. With the two-photon memory, all pages have to be read out for this operation for the given selectivities. Thus, the times for the two-photon memory only reflect the superior raw bandwidth and not the data isolation features of the bi-orthogonal accessing approach. The superior performance is clear. The performance improvement should be even more notable for operations utilizing the bi-orthogonal accessing capability.

			3D two-photon memory	
	PHOEBUS	GAMMA (using 30 of its 32 processors)	(record parallel packing)	(field parallel packing)
memory device	single parallel access optical disk	30 sequential access disks	3D 2-photon memory ($w = 2^5$ bits)	3D 2-photon memory ($w = 2^5$ bits)
secondary storage bandwidth	208*8/240nsec = ~ 9 Gbits/sec	32/() = 200 Mbits/sec (pg 58)***	(1024 x 1024)/ 10 usec = 200 Gbits/sec	(1024 x 1024)/ 10 usec = 200 Gbits/sec
1% of records	.42 sec	8.16 sec	.0164 sec	.0161 sec
10 % of records	.42 sec	10.82 sec	.0164 sec	.0161 sec

Table 4-5: Times to perform selection with no indexing for GAMMA, PHOEBUS, and the bi-orthogonally accessed two-photon memory based data filter assuming a relation with 10^6 208 byte records

5. FEASIBILITY STUDY

The performance potential of the bi-orthogonally accessed 3D memory lead to a study to determine the feasibility of building an optoelectronic database data filter to interface with this memory. In the sections that follow, components that would be needed for a data filter are described in terms of functionality, speed, packaging requirements, area and power. Issues relating to the implementation of the actual bi-orthogonally accessed 3D two-photon memory are not discussed.

5.1 System description overview

Figure 5-19 shows the components of the data filter that were considered in this study with their assumed data rates and formats. Figure 5-20 illustrates details of the packaging requirements. It is assumed that that w = 8 bits; this choice has little effect. In the discussions that follow, it is assumed that fields are a multiple of w bits in size.

The optoelectronic data filter consists of a modified CCD array called a compander which is flip-chip bonded onto an electronic processing chip capable of performing data filtering operations. The number of bits on a page retrieved from the 3D two-photon memory, M^2, is large, approximately 10^6. The memory page rate, f_{opt} ,is likely not to be that fast. 100 Kpages/sec is assumed in this study. This is a slow speed and a large amount of data for an electronic processing chip. The compander [15] functions to convert the data retrieved from the parallel readout optical memory into a format more compatible with electronic processing, i.e., a higher data rate and reduced parallelism.

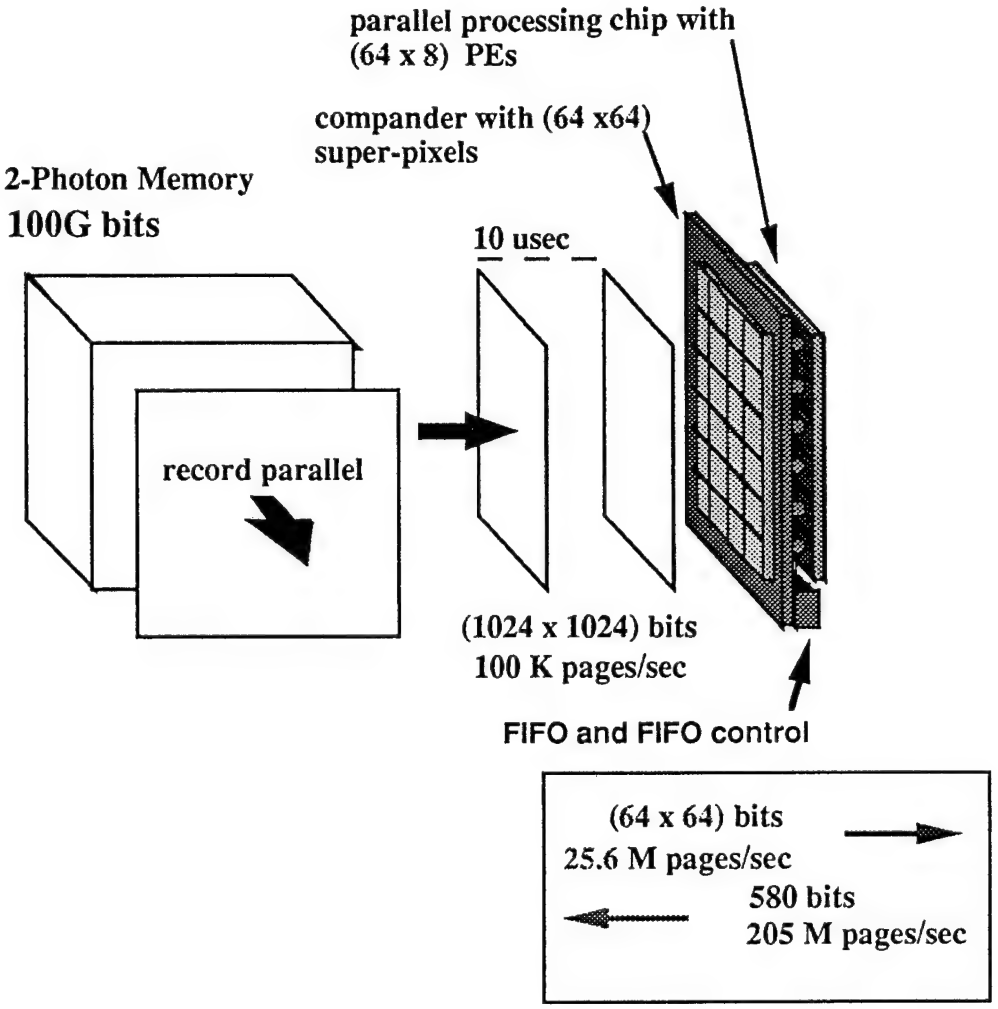

Figure 5-19: Components that were assumed for the database data filter with their data rates.

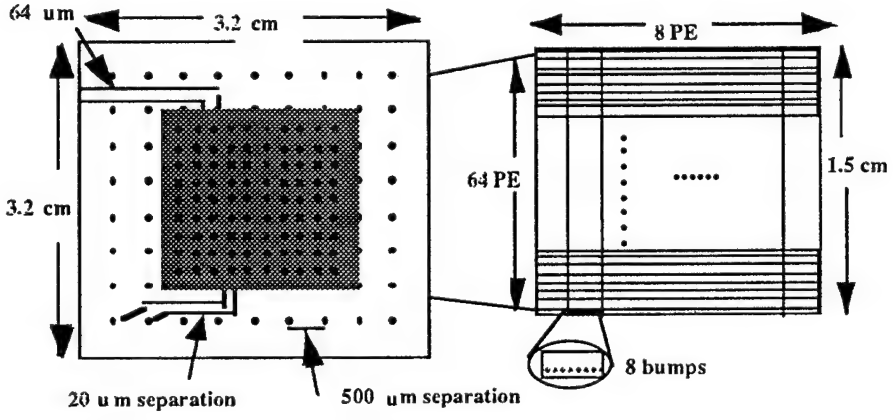

Figure 5-20: back side of compander and processing chip (left) and front side of processing chip (right).

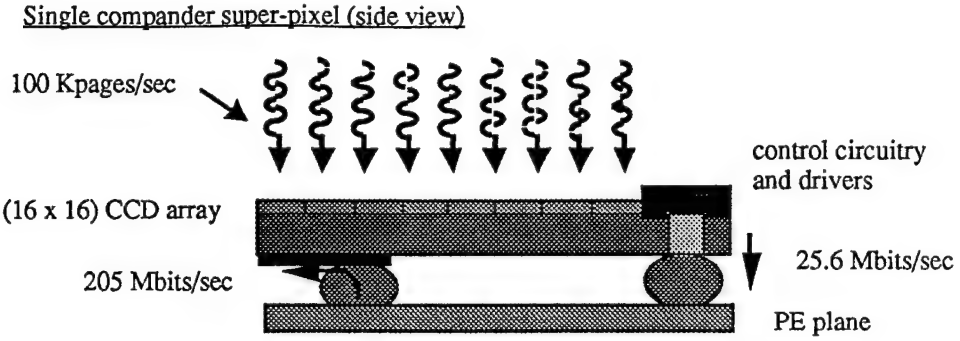

Figure 5-21: Side view of a compander super-pixel

The compander is shown in Figure 5-19 and Figure 5-21. The M^2 CCD sensors on the compander are partitioned into $(M/n)^2$ super-pixels. Each super-pixel has an array of n^2 CCD sensors, some control circuitry, a driver, a via and a flip-chip solder bump. In operation, the compander detects a page of M^2 bits from the memory. Then simultaneously each super-pixel in the compander serially rasters out its contents (amplified) through its via to the front side of the processing chip. This procedure has the effect of converting a single memory page of M^2 bits transmitted at a rate f_{opt} into n^2 smaller pages, referred to as a *compander pages,* which it sends to the processing chip a rate $n^2 f_{opt}$. Each compander page would have $(M/n)^2$ pixels. The bandwidth into and out of the compander is the same, thus no bottlenecks exist; just the data format is altered.

Fabrication limitations constrain the vias on the compander to be at least 500 um apart [18]. This constraint led to choosing super-pixels with 256 CCD sensors (n = 16). There would be approximately 4,000 super-pixels. The resulting compander would be 3.2 cm x 3.2 cm in size, and the compander page rate would be 25.6 MHz.

The processing chip is estimated to be 1.5 cm x 1.5 cm in size and has $(M/n)^2$ solder bumps to obtain data from the compander. (Details of this chip will be discussed in a following section.) Because of the compander and processing chip size mismatch, the solder bumps on the chip are not directly underneath the compander vias that they are affiliated with. The back side of the compander chip is used as an interconnection plane to route the signals from the compander vias to solder bumps on the chip as shown in Figure 5-20.

The processing chip itself has (64 x 8) = 512 PEs each with eight solder bumps that receive data from the compander. The data, after being, processed by the chip is returned to the back side of the compander through another set of solder bumps (~ 580 of them) at a rate of 205 MHz. Two metal layers on back side of the compander would necessitate 20 um and 65 um metal line separation for the lines into and out of the chip, respectively.

5.2 Data format and the compander

In operation the compander de-interleaves (in two dimensions) a data page from memory. To compensate for this, a word desired in parallel by the processing chip must be spatially interleaved on a memory page with other words which would arrive at the processing chip in parallel at a different time. Since words are desired in parallel from both orthogonal directions,

they should be interleaved in only one-dimension as opposed to two, and interleaved to a depth of n.

Figure 5-22 depicts how records could be placed in the memory to compensate for the transformation performed by the compander.

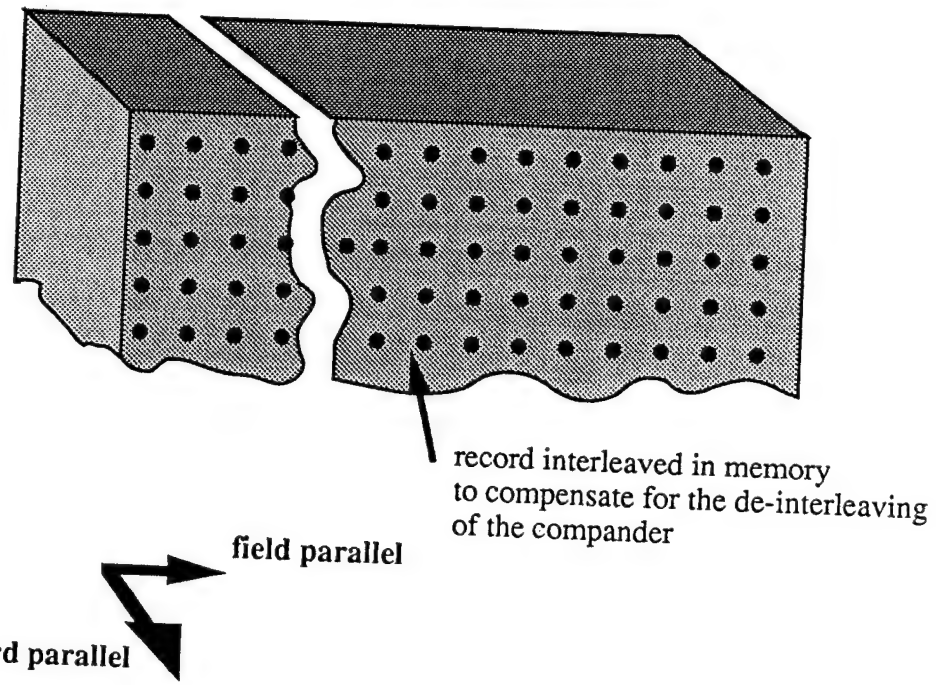

Figure 5-22: record interleaved in memory to compensate for the de-interleaving of the compander

Temporal interleaving occurs in the system as well. Each field parallel page contains approximately 130 Kbytes of information, and each byte belongs to a separate record. The data on each field parallel page is arranged (interleaved) so that each of the 512 PEs receives an eight bit word in parallel belonging to a unique record. Each PE receives n^2 such words -- one for each of the compander pages that a single memory page in converted into. Since each of these words belongs to a unique record, from the processing chip's point of view, fields retrieved with field parallel access are interleaved in time, to a depth of n^2, with data belonging to the same field but to other records. The 55 Mbit FIFO with control is used to perform the temporal de-interleaving. The FIFO is large enough to store the largest field of the Wisconsin benchmark that would be contained in a single block. Current SRAM technology provides enough bandwidth so that a bank of SRAM chips could both store and readout records simultaneously.

5.3 Processing chip functionality

The processing chip as mentioned earlier is partitioned into 512 PEs. Each has a digital comparator and some memory. The functionality of the chip and its hardware are described by explaining how two operations would be performed: selection and selection-projection with no

indexing with field-record or field-field access. These are the most complex operations to implement on the chip. The other operations are just mentioned.

5.3.1 Example: selection and selection-projection with no indexing

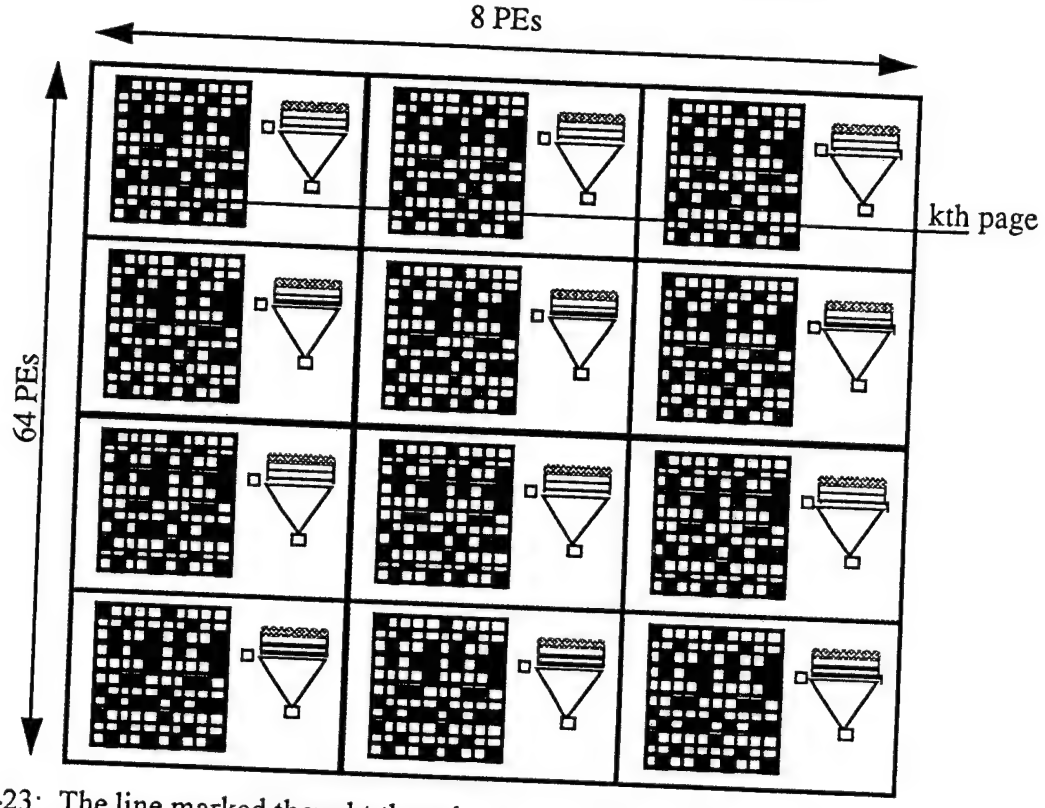

Figure 5-23: The line marked thought the select status bit memories indicates which select status bits correspond to selected records on the kth page in the field parallel direction. The select status bits in each PE are written left-to-right (or right-to-left); when a row is filled a new one either directly above or below is started.

With these two operations, one must first search through all the records to determine which ones satisfy the selection query. Field parallel access is used so that only the operand containing field would be retrieved. Records identified as satisfying the selection query would then be read out using field or record parallel access.

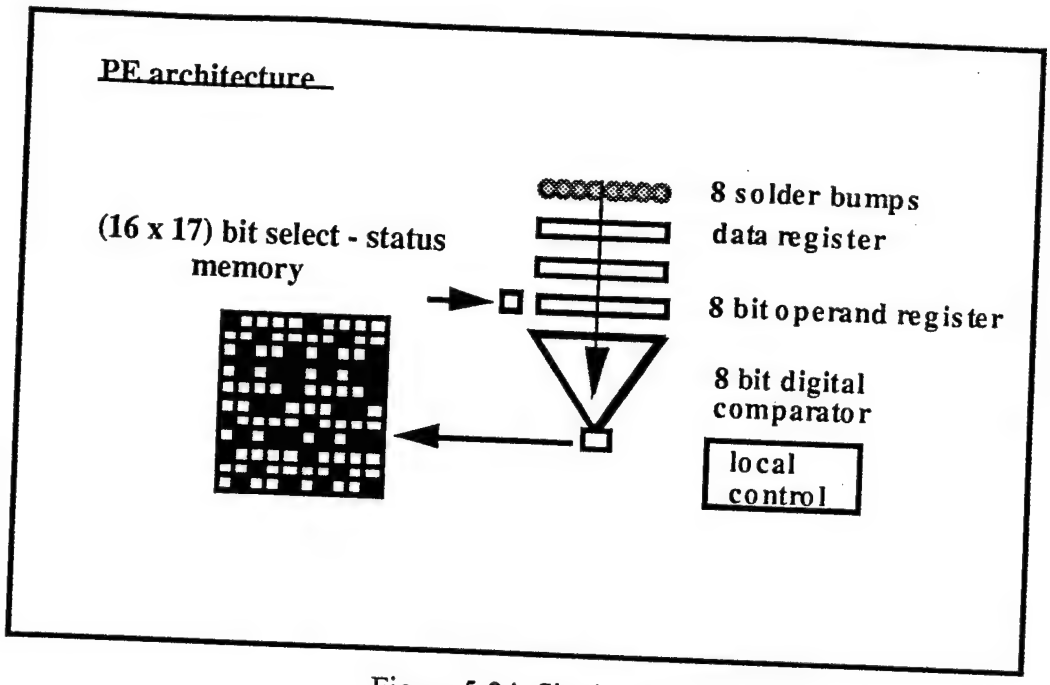

Figure 5-24: Single PE

The 512 processing elements (PE)s, are roughly illustrated in Figure 5-23 and Figure 5-24. In the search part of the operation, each PE initially receives the first 8 bits of the field that is being examined. It performs the desired comparison on this word with a locally stored operand. The result of the comparison is stored in the first bit of its n^2 bit *select-status* memory. Since records retrieved with field parallel access are interleaved in time to a depth of n^2, each PE receives n^2-1 more bytes (each byte belonging to a unique record). It makes n^2-1 similar comparisons with the same locally stored operand. The results of these comparisons are stored in the remaining n^2-1 bits of the select-status memory. While the above described comparisons are being performed on the data from the first memory page, the second byte of the operand is daisy-chained, with a single daisy chain, to each PE at a rate of 205 MHz. At this speed, all operands can be loaded on chip while the first memory page is being examined and will be in place when data from the next memory page arrives. A single daisy chain allows each operand to be unique. Although not important for this operation, this feature is useful for operations performed with record parallel access. After all the operands have been loaded, they are transferred to the operand registers in each PE in time to be compared with the data from next page of the memory. The results of the comparisons on this second byte are combined with those from the previous byte (stored in the select status memory) and restored in the select-status memory. The procedure is repeated for as many bytes as the field has. In the end the bits in the select status memories reflect which records in a single block have been selected.

The bits in the select status memories are used to identify selected records when they are read from the memory at a later time. For field parallel access this is trivial with the data organization considered in this study. The bits in each select status memory can be read out in the order that they were stored to directly mask out fields not belonging to selected records as they arrive. If every record is selected on a given compander page, there is enough bandwidth to

send all results along with their select-status bits off-chip before the next compander page arrives.

For record parallel readout, the select status bits are used to determine which pages in the record parallel orientation have selected records on them and will need to be read out. These bits are not used to indicate the exact position of the selected bits on the record parallel page; the selection operation is just repeated. Figure 5-23 shows the correspondence between bits in the select-status memories and pages in the record parallel direction, assuming that bits are written into these memories in the order indicated. (The compander super-pixel read out must also mimic this read out topology.) If any bit in the kth row of the combined select status memories is ONE it would indicate that the kth page in the record parallel direction has at least one selected record on it. In an earlier section it was mentioned that this operation is best performed by searching all blocks in each super-block using field parallel access and then reading out selected records in the record parallel direction. This can efficiently be implemented with the described hardware. The n^2 values in the select-status memory determined for one block or several blocks of information can be ORed with results from the next block as they are determined. When the final block of the super-block is being checked for selectivity, bits in the select-status memories are ORed together to determine which pages in the record parallel direction need to be read out.

As mentioned, the selections need to be performed a second time when the data is retrieved with record parallel access. If the selectivity results of each block were not combined much more memory on the chip would be needed to store them. For the Wisconsin benchmark the chip area would grow by approximately a factor of four which seemed unreasonable.

If it is determined that a page in the record parallel direction has a selected record on it, operands can be appropriately positioned on the chip with the daisy chain register so that the desired comparisons can be performed. Alternatively these comparisons can be done off-chip. The masking and comparison operations required for all other data filtering operations can likewise be performing either on or off-chip.

5.4 Chip area

Table 5-6 and Table 5-7 list the assumptions used to estimate the chip area assuming a .8 um MOSIS process. With these values, the chip circuitry was determined to occupy .72 cm x .72 cm. The area required for clock distribution, global control and for interconnects is not included in this estimate. This circuitry area estimate was multiplied by 10/3 to allow a bit more than 70% of the area for these features. This is generous since the chip is essentially a memory chip.

component	area
1 bit minimum sized register	15 um x 15 um
8 bit digital comparator *	4800 um^2
local decoder for each PE	30 um x 30 um
solder bump	25 um x 25 um
output buffer	125 um x 125 um

* area estimated from reference 19

Table 5-6: Assumptions that were made to estimate the area required for the processing chip.

	Area
Chip circuitry	39.2 mm^2 = 6.3 mm x 6.3 mm
output buffers	580 $(125\ um)^2$ = 9.1 mm^2
solder bumps from Compander	4096 $(25\ um)^2$ = 2.6 mm^2
I/O solder bumps	1000 $(25\ um)^2$ = .6 mm^2
Total chip area	~52 mm^2 = 7.2 mm x 7.2 mm

Table 5-7: Summary of area required for processing chip.

5.5 Chip power

The parameters used to estimate the power for the chip are listed in Table 5-8 assuming two phase clocking. The power required for the registers and the inverters was determined by SPICE simulation. The estimate for the chip power is broken down as follows:

$$P_{CHIP} = P_{chip\ clk} + P_{circuitry} + P_{off\text{-}chip\ drivers}$$

$P_{chip\ clk}$ is the total power required for the on-chip clock; $P_{circuitry}$ is the power required for the circuitry; and $P_{off\text{-}chip\ drivers}$ is the power required to send the data off-chip.

Parameters used	description	value
D_{CCD}	linear compander dimension	3.2 cm
D_c	linear chip dimension	1.5 cm
M^2	memory page size	1024^2 bits
$(M/n)^2$	number of super-pixels	~ 4K bits
f_c	chip clock speed	205 MHz
f_{opt}	memory page rate	100 K pages/sec
$n^2 f_{opt}$	Compander page rate	25.6 MHz
	power for 1 bit register @ 205 MHz	1 mWatt
	power for 1 bit register @ 25.6 MHz	1/8 mWatt
N_{oput}	number of output solder bumps @ 205 MHz	~580
C_g	gate capacitance	3 fF
c_i	metal capacitance	.02 $fF/(um)^2$
V_{DD}	power supply voltage	3.3 Volts
w_i	width of interconnects	3 um
$P_{clk\ B}$	static power dissipation for a minimum sized buffer for the clock tree	.5 mWatts
C_b	capacitance of a solder bump	1 pF
$P_{off\text{-}chip\ driver}$	static power dissipation for a single off-chip driver	34.5 mWatts

Table 5-8: Parameters used to estimate the power required for the processing chip.

5.5.1 Power for chip clock distribution

The clock is assumed to be distributed with a H-clock tree. The equation used to estimate the power needed for the on-chip clock, $P_{chip\ clk}$, is given below. N_{leaf} is the number leaves in the clock tree and N_g is number of gates operating at 205 MHz at each leaf that receive the clock signal. The first term in the bracketed part of the equation is the power to drive the gates on-chip with the clock; the second term is the power required to drive each interconnect in the clock tree; and the third term is the power needed for the buffers in the clock tree. The capacitance of the clock tree interconnect was estimated using reference 20. Because of the clock speed, it was assumed that the buffers in the clock tree would be operating in a linear regime and thus would predominately experience static power dissipation: $P_{clk\ B}$. It was also assumed that the second clock could be generated from the first at the base of the clock tree.

$$P_{chip\ clk} = N_{leaf}\,(1/2\,(N_g\,C_g + c_i\,w_i\,D_c)\,V^2_{DD}\,f_c + P_{clk\ B})$$

where, $N_{leaf} = (M/n)^2$ and $N_g = 9$.

5.5.2 Power for circuitry

The average peak power for the circuitry was estimated and determined to be approximately 10 Watts. Not all of the components would be operating at once, and some only for a brief period of time. This was taken into consideration.

5.5.3 Output driver power

The average power required to send the data off-chip was estimated with the equation below. As with the clock tree calculation, it is assumed that the off-chip drivers and the buffers used to drive the drivers (included in the calculation for $P_{off\text{-}chip\ driver}$) would predominantely experience static power dissipation. The capacitance term reflects the power needed to drive the off-chip lines and the solder bumps, respectively, assuming that on average half the buffers would be switching at a given time.

$$P_{off\text{-}chip} = (1/2)\,N_{oput}\,(P_{off\text{-}chip\ driver} + 1/2(c_i\,w_i\,D_{CCD}/2 + C_b)V^2_{DD}\,f_c)$$

The power requirements for the chip are summarized in Table 5-9.

	Power (Watts)
Chip circuitry	~10 Watts
Chip Clock	~6 Watts
Power for chip output	~21 Watts
Total chip power	~37 Watts

Table 5-9: Summary of estimate for the power for the processing chip.

5.6 Compander power

Table 5-10 lists the parameters that were used in the power calculation for the compander along with their assumed values. The total power required for the compander, P_{cmp}, is estimated by:

$$P_{cmp} = P_{CCD\ clk} + P_{cmp\ ctl} + P_{drivers}$$

$P_{CCD\ clk}$ is the power for the clock on the CCD; $P_{cmp\ ctl}$ is the control power for the CCD which causes the pixels in each super-pixel to be read out in the correct order; $P_{drivers}$ is the electrical power need to amplify the detected signals and to send them to the processing chip. The compander was assumed to have three phased clocking.

CCD parameters	description	value
η	quantum efficiency of CCD sensors	50%
D_{CCD}	linear CCD dimension	3.25 cm
$C_{g\ CCD}$	capacitance for single CCD pixel	10 fF
w_i	width of interconnect lines	3 um
$P_{CCD\ ctl}$	electrical control power for each super-pixel	negligible
d_{via}	distance between vias on Compander	500 um
C_{via}	compander via capacitance	1 pF
C_b	solder bump capacitance	1 pF
$P_{C\ clk\ B}$	static power dissipation of buffer in clock tree	.5 (1/8) mWatts
P_{amp}	static power dissipation of amplifier in each super-pixel	.1 mWatts
c_i	metal capacitance	.02 fF/(um)^2
V_{DD}	power supply voltage for compander	3.3 Volts
λ	wavelength of detected light	640 nm
P_{opt}	optical power per single bit	10 nWatts [21]

Table 5-10: Parameters used to estimate the power for the compander.

5.6.1 Power for compander control circuitry

The power for required for the total control circuitry was determined using the power for a single bit register operating at n f_{opt}. Because of the slow operational speed, $P_{cmp\ ctl}$ was found to be very small and was neglected.

5.6.2 Power for compander clock distribution

The clocks signals on the compander were also assumed to be distributed with an H-tree. There are two three phased clock systems: a slow clock and a fast clock. The slow clock operates at a rate: nf_{opt} and is responsible for shifting the "rows" of each super-pixel into faster n bit CCD arrays for eventual output. The faster CCD array operates at the compander page rate. All M^2 CCD sensors are driven by the slower clock. M^2/n additional CCD sensors are driven by

the faster clock. It is assumed that one of the three clocks in the three phase clocking scheme can be generated at the leaf nodes for both the slow and fast clocking systems. The equation for the power required for clock distribution is thus:

$$P_{CCD\,clk} = (N_{leaf\,fast} + N_{leaf\,slow}/n)\,(P_C\,Clk_B + 3/2\,(C_{g\,CCD} + (2/3)\,c_i\,w_i\,D_{CCD})\,V^2_{DD}\,f_{cmp})$$

where, $N_{leaf\,fast} = M^2/(3n)$ and $N_{leaf\,slow} = M^2/3$.

5.6.3 Power to send data to the processing chip

It was first verified that the optical power per bit would generate enough electrons to be reasonably detected. With the variables in Table 5-10, the number electrons that are generated per bit can be expressed as:

$$N_e = \eta\,P_{opt}/(f_{opt}\,h\,\nu)) = 160\ K$$

this corresponds to the following voltage on each CCD gate:

$$V_{detection} = (1.6\ X\ 10^{-19}\,N_e)/C_{g\,CCD} = 2.56\ Volts$$

This voltage could easily be amplified to the rail voltage. The power needed to amplify the detected signals and send them off chip is estimated with the equation below. The first term reflects the static power dissipation of the amplifier. The second term accounts for the power needed to drive the vias, the solder bumps and the lines routing the data from the compander to the processing chip.

$$P_{drivers} = (M/n)^2\,P_{single\ driver}$$

where, $P_{single\ driver} = (M/n)^2(\,P_{amp} + 1/2\,(C_{via} + C_b + (1/2)\,c_i\,w_i\,D_{CCD})\,V^2_{DD}\,f_{cmp})$

The power breakdown for the compander is summarized in Table 5-11. The power required for the entire data filter would be approximately 50 Watts.

	Power (Watts)
Compander Clock	~9 Watts
Power to amplify signal and send data to chip	~2 Watts
Total compander power	~11 Watts

Table 5-11: Summary of power required for the compander.

6. CONCLUSIONS

The 3D two-photon memory, in general, was found to be well suited for very large databases with low write requirements because of its potential capacity, raw bandwidth and random access capability. It was determined that performance can be significantly improved by utilizing an accessing feature termed bi-orthogonal access, a feature which allows both record and field parallel access to be supported simultaneously. It was noticed that the way that records

are placed in the memory affects performance. No particular record placement/packing strategy was found to be clearly advantageous for all operations. In a system this choice would have to be made by anticipating frequent operations. For the relation size that was considered in this study (10^6 records) the packing strategy was found to have greater impact than the effect of memory fragmentation. This effect of packing would reduce with larger relation sizes, but would increase with smaller sized relations.

A metric termed bit retrieval efficiency was devised to evaluate the optimality of relational database operations performed with a bi-orthogonally accessed 3D optical memory. The results of this analysis indicate that the throughput and the random access capability of a memory both affect the performance of relational database machines. A 3D two-photon memory with bi-orthogonal access can viewed as having improved random access, and in this way improved performance. A shortcoming that all massively parallel access optical memories have, in terms of this application, is that an entire page of records has to be retrieved even if only one record is desired. From this vantage they appear to have reduced random access capabilities. A bi-orthogonally accessed 3D optical memory with the same bandwidth but with a reduced page size would have improved performance.

7. REFERENCES

1. D. J. DeWitt, S. Ghandeharizadeh, D. A. Schneider, A. Bricker, H. Hsiao, and R. Rasmussen, "The Gamma Database Machine Project," IEEE Trans on Knowledge and Data Engineering 2, pp. 44-61 (1990).
2. http://www.oracle.com/info/news/parallel.html
3. S. Hunter, F. Kiamilev, S. Esener, D. A. Parthenopoulos, and P. M. Rentzepis, "Potentials of 2-photon based 3-D optical memories for high performance computing," Applied Optics **29**, pp. 2058-66 (1990).
4. P. A. Mitkas and P. B. Berra, "PHOEBUS: An optoelectronic database machine based on parallel optical disks", Journal of Parallel and Distrib. Comp. **17**, pp. 230-44 (1993)..
5. G. W. Burr, F. H. Mok, and D. Psaltis, "Storage of 10,000 holograms in $LiNbO_3$: Fe," CLEO '94 Anaheim, CA, p. 9 (1994).
6. I. Cokgor, F. B. McCormick, A. S. Dvornikov, P. M. Rentzepis, S. C. Esener, and M. M. Wang, "Operation and properties of 2-photon based 3-dimensional optical memory systems," Accepted by Applied Optics pending revisions.
7. P. A. Mitkas and L. J. Irakliotis, "Three-dimensional optical storage for database processing," Optical Memory and Neural Networks **3**, pp. 217-229 (1994).
8. D. J. DeWitt, "The Wisconsin Benchmark: Past, Present and Future" In *The Benchmark Handbook*, report 4, Ed. Jim Gray, Morgan Kaufmann, San Mateo (1993).
9. G. Gardin and P. Valduriez, *Relational Databases and Knowledge Bases,* Report 4, Addison-Wesley Publishing Co., Menlow Park, CA (1989).
10. G. Gardin and P. Valduriez, *Relational Databases and Knowledge Bases,* Report 9, Addison-Wesley Publishing Co., Menlow Park, CA (1989).
11. R. A. Athale and M. W. Haney, "Optical implementation of numerical inequality detection and its application to database machines," Optics Letters **17**, pp. 1611-13 (1992).
12. P. A. Mitkas, L. J. Irakliotis, F. R. Beyette Jr., and S. A. Feld et al., "Optoelectronic data filter for selection and projection," Applied Optics **33**, pp. 1345-53 (1994).
13. S. Y. W. Su, *Database Computers*, Report 4, McGraw-Hill, San Francisco, CA (1988).
14. D. Schneider and D. J. DeWitt, "A performance evaluation of four parallel join algorithms in a shared-nothing multi-processor environment", Proc. 1989 SIGMOD Conf., Portland, OR (1989).
15. Private conversation with Dr. Sadik Esener
16. *The Benchmark Handbook*, Ed. Jim Gray, Morgan Kaufmann, San Mateo (1993).
17. S. Hunter, "Three-Dimensional Optical Memory Systems Based on 2-Photon Excitation: System Studies and Component Design," Dissertation: University of California, San Diego (1994).

18. Private conversation with Dr. Philippe Marchand
19. N. H. E. Weste and K. Eshraghian, *Principles of CMOS VLSI Design : a Systems Perspective*, Addison-Wesley, Reading, MA (1988).W&E
20. H. B. Bakoglu, Circuits, Interconnections, and Packaging for VLSI, Addison-Wesley Pub. Co., Reading, MA (1990).
21. Private conversation with Dr. Rick McCormick.